Efficient Quantum Simulation

Euan Allen

Advanced Quantum Information Theory Essay

Quantum Engineering Centre for Doctoral Training, The University of Bristol

April 9, 2015

1 Introduction

Quantum simulation, the idea that you can simulate a quantum system using another quantum system, is thought to be one of the main applications of quantum computation. Feynman was the first to suggest the this might be useful, given the difficulties of performing quantum simulations on classical machines [1]. Simulating the time evolution of an arbitrary quantum system is intractable for a classical machine [2]. In other words, simulating a quantum system on a classical computer scales exponentially with the complexity of the system. We can see this by considering a collection of n two-level systems (qubits). To record the state of the system we need to store 2^n complex numbers. To calculate the time evolution of this system, we need to exponentiate a $2^n \times 2^n$ matrix. For a relatively modest number of qubits, this number is very large (e.g. $n = 40$ is a matrix of $\sim 10^{24}$ entries). A 'back of the envelope' calculation by Brown *et al.* [3] shows that for a modest $n = 27$, to store the complex numbers needed to define a quantum state would require $\sim$ 1gb of memory. One of the largest classical simulations of a quantum system to date required 1tb of memory, 4096 processors and involved the simulation of 36 qubits [4]. It is unlikely that there will ever be an efficient classical algorithm for simulating the dynamics of quantum systems. The problem is BQP-complete, and so an efficient classical algorithm for quantum simulation would mean that any problem that is efficient for a quantum computer could be done efficiently on a classical one (e.g. Shor's factorisation algorithm) [5].

There are two general approaches that can be taken with respect to quantum simulation. The first, often referred to as 'analogue quantum simulation', is a simulation of a quantum system by use of another system with similar dynamics. In this approach, the Hamiltonian of the system you are trying to simulate, $\mathcal{H}_{sys}$, is mapped onto the Hamiltonian of the simulator, $\mathcal{H}_{sim}$, which you have more control of. In essence the simulator system is used to mimic the first (sometimes also referred to as a 'quantum emulator'). The requirement of similar dynamics between the two systems restricts the class of problems that any analogue simulation can emulate. It is possible though that analogue quantum simulation may be the easier of the two to implement experimentally [3].

The second approach, sometimes referred to as 'digital quantum simulation', is the application of a quantum algorithm to suit the model of your physical system. In this method, you encode the quantum state of your system into qubits and then 'evolve' the system by application of a set of unitary gates. It is this method of approach that will be the focus for the rest of the essay.

In quantum mechanics, the evolution of the state of a quantum system, $|\psi\rangle$, is governed by the Schrödinger equation:

$$i\hbar \frac{d}{dt} |\psi(t)\rangle = H(t) |\psi(t)\rangle \, , \tag{1}$$

where $H(t)$ is the Hamiltonian of the quantum system. For a time-independent system where $H(t) = H$ is constant, the solution of the Schrödinger equation is

$$|\psi(t)\rangle = e^{-i\hbar Ht} |\psi(0)\rangle \, . \tag{2}$$

It is the aim of quantum simulation to approximate this evolution. Specifically, we aim in quantum simulation to approximate the unitary

$$U(t) = e^{-i\hbar Ht} \tag{3}$$

to within a certain error. Note that from now on we will ignore factors of $\hbar$. Typically, a unitary U' is said to approximate the unitary U to within ϵ if

$$\|U' - U\| \leq \epsilon, \tag{4}$$

where the operator norm (also known as the spectral norm or induced Euclidean norm) is defined as:

$$\|A\| \equiv \max_{|\psi\rangle \neq 0} \frac{\|A |\psi\rangle\|}{\||\psi\rangle\|}, \tag{5}$$

and $\||\psi\rangle\| = \sqrt{\langle\psi|\psi\rangle}$. For Hermitian matrices, $\|H\|$ is equal to the magnitude of the largest eigenvalue of that matrix.

An important point to make is that these methods of Hamiltonian simulation are not only useful for simulating actual physical systems, but can also be used to implement quantum algorithms that can be defined in terms of Hamiltonian dynamics (for example continuous-time quantum walks and adiabatic quantum computing).

2 Quantum Simulation Algorithms

There have been a number of different algorithms proposed for the simulation of quantum systems. Almost all proposals have been limited to the solution of one of two classes of Hamiltonians: k-local and d-sparse (the simulation of non-sparse Hamiltonians has been looked in to by Childs and Kothari [6]). A Hamiltonian acting on n qubits is said to be k-local if it can be written in the form

$$H = \sum_{j=1}^{l} H_j, \tag{6}$$

for some $l \leq \binom{n}{k}$, where H_j is a Hermitian matrix acting on a space of at most 2^k dimensions. There are many physical systems that follow these requirements: hard-sphere and van der Waals gases, Ising and Heisenberg spin systems, lattice gauge theories, and so on [2].

A Hamiltonian of dimensions $N \times N$ is said to be d-sparse (in a fixed basis) if there are at most d non-zero entries per row, where $d = \text{poly}(\log(N))$. One can note that a k-local Hamiltonian with l terms (see Equation (6)) is d-sparse with $d = 2^k m$ [7]. This means that any algorithm capable of simulating a sparse Hamiltonian is also capable of simulating a local one.

We will now take a detailed look at three different methods of quantum simulation proposed over the years, starting off with the first real attempt at a digital quantum simulation proposed by Lloyd.

2.1 Trotter Decomposition - Lloyd (1996)

The first attempt at a quantum algorithm from Feynmans first postulate was completed by Seth Lloyd in an attempt to simulate k-local Hamiltonians [2].

Theorem 2.1. (Solovay-Kitaev Theorem). *Let U be a unitary operator which acts non-trivially on k qubits, and let S be an arbitrary universal set of quantum gates. Then U can be approximated in the operator norm to within ϵ using $O(\log^c(1/\epsilon)$ gates from S, for some $c < 4$. The value of c varies between different proofs of the theorem. For example Dawson and Neilson give a value of $c \approx 3.97$ [8].*

Interestingly, this theorem tells us that a unitary operator that can efficiently realised with a universal set of quantum gates can also be realised efficiently with another such set to within a bounded error. More precisely; the running time of an algorithm only varies by a logarithmic factor between two universal gate sets meaning that polynomial quantum speedups are robust against the choice of gate set [8].

Lemma 2.2. (Lie-Trotter Product Formula). *Let H_1 and H_2 be Hermitian matrices such that $\|H_1\| \leq K$ and $\|H_2\| \leq K$, for some real $K \leq 1$. Then*

$$e^{-iH_1}e^{-iH_2} = e^{-i(H_1+H_2)} + O(K^2). \tag{7}$$

Repeated application of this formula for multiple Hermitian matrices $H_1, ..., H_l$ all satisfying $\|H_j\| \leq K \leq 1\ \forall j$, allows one to write the following relation:

$$e^{-iH_1}e^{-iH_2}...e^{-iH_l} = e^{-i(H_1+H_2+...H_l)} + O(l^3K^2). \tag{8}$$

We can see from our definition of a k-local Hamiltonian (Equation (6)) that what we would like to simulate is the exponent of a sum of Hermitian operators

$$U = e^{-iHt} = e^{-i \sum_{j=1}^{l} H_j t}. \tag{9}$$

Using Lemma 2.2, one can show that for a constant C such that $n \geq Cl^3(Kt)^2/\epsilon$

$$\| e^{-iH_1 t/n} e^{-iH_2 t/n} ... e^{-iH_l t/n} - e^{-i(H_1 + H_2 + ... H_l)t/n} \| \leq \epsilon/n. \tag{10}$$

Using the result of Lemma 2.3, we can re-write Equation (10) as

$$\| (e^{-iH_1 t/n} e^{-iH_2 t/n} ... e^{-iH_l t/n})^n - e^{-i(H_1 + H_2 + ... H_l)t/} \| \leq \epsilon. \tag{11}$$

The Solovay-Kitaev theorem tells us that it is possible to approximate each $e^{-iH_j t}$ individually to within an error of ϵ in time $O(\text{polylog}(1/\epsilon))$. This fact along with Equation (11) shows us that it is possible to approximate the operator e^{-iHt} to within ϵ in time $O(l^3(Kt)^2/\epsilon)$, up to polylogarithmic factors. By redefining $K = \|H\|$ and using that $l = O(n^K)$ where $N = 2^n$, then we can see that this algorithm scales as $O(\text{polylog}(N)(\|H\|t)^2/\epsilon)$ (see Table 1).

Lemma 2.3. *Let* (U_i), (V_i) *be sequences of* m *unitary operators satisfying* $\|U_i - V_i\| \leq \epsilon \; \forall i$ *where* $1 \leq i \leq l$. *Then* $\|U_i U_{i-1} ... U_1 - V_i V_{i-1} ... V1\| \leq l\epsilon$.

This algorithm can be quite simply improved upon by a slight re-ordering of the exponents as per the Lie-Trotter-Suzuki formulae:

$$(e^{-iAt/n} e^{-iBt/n})^n = e^{-i(A+B)t} + O(t^2/n), \tag{12}$$

$$(e^{-iAt/2n} e^{-iBt/n} e^{-iAt/2n})^n = e^{-i(A+B)t} + O(t^3/n^2), \tag{13}$$

$$\vdots$$

where expansions to arbitrary order are known [9]. Work has been done to show that the k^{th} order expansion with an error upper bound of ϵ, requires at most

$$5^{2k} m^2 \|H\| t \left(\frac{m\|H\|t}{\epsilon} \right)^{1/2k} \tag{14}$$

exponentials [10].

2.2 Sparse Matrix Simulation - Aharonov & Ta-Shma (2003)

The first instance of an algorithm to simulate a sparse matrix was completed by Aharonov and Ta-Shma in 2003 [11]. In order to show how this is done, we first need to introduce the following definitions:

Definition (Row Computability) A matrix H is said to be row computable if given a row index i, there exists an efficient algorithm to output a list $(j, H_{i,j})$ running over all non-zero entries in the row i. This is often thought of as a query to a black-box that given the input i, will output the list of non-zero entries in the row i $((j, H_{i,j}))$.

Definition (Combinatorial Block Diagonal Matrix) A block diagonal matrix (in the traditional sense) is a matrix that can written in the form

$$\begin{pmatrix} \mathbf{M_1} & 0 & \cdots & 0 \\ 0 & \mathbf{M_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{M_N} \end{pmatrix}$$

where each $\mathbf{M_i}$ is a square matrix of arbitrary dimensions. A matrix is said to be a $n \times n$ block matrix if the matrices $\mathbf{M_i}$ are at most $n \times n$ in size. A matrix is *combinatorially* block diagonal if under perturbation of rows or columns, it is possible to construct a block diagonal matrix in the usual sense (this definition is stated more precisely in Definition 6 of [11]).

The outline of the argument in [11] is as follows: First it is shown that it is possible to decompose H into a superposition of 2×2 combinatorially block diagonal matrices (Lemma 2.4). Next it is proven that each of these matrices can be efficiently simulated (Lemma 2.5), much like the Solovay-Kitaev Theorem in Section 2.1. Finally Trotter decomposition is used (as before) to show that the total Hamiltonian, H, can be efficiently simulated. The following text will prove Lemma 2.4.

Lemma 2.4. (Decomposition Lemma) *Let H be a d-sparse, row-computable Hamiltonian over n qubits. It is possible to decompose H into $H = \sum_{m=1}^{(d+1)^2 n^6} H_m$ where each H_m is:*

- *A sparse, row-computable Hamiltonian over n qubits, and,*
- *A 2×2 combinatorially block diagonal.*

Proof. For the Hamiltonian H, we label each entry of the matrix $H_{i,j}$ for $i < j$ (upper diagonal entries) by a colour[1]. The colour of an entry $col_H(i,j)$ is defined as the tuple $(k, i\text{mod}k, j\text{mod}k, r_H(i,j), c_H(i,j))$ where

$$k = \begin{cases} 1 & \text{if } i = j \\ \text{The lowest integer where } i \neq j mod k \text{ for } 2 \leq k \leq n^2 & \text{otherwise,} \end{cases} \tag{15}$$

and

$$r_H(i,j) = \begin{cases} 0 & \text{if } H_{i,j} = 0 \\ \text{The index of } H_{i,j} \text{ in the list of all} & \\ \text{non-zero elements in the } i^{th} \text{ row of H} & \text{otherwise.} \end{cases} \tag{16}$$

The definition of $c_H(i,j)$ is the same as Equation (16) but now for columns as opposed to rows. For values of $i > j$ (lower diagonal entries), we define

[1]The 'colour' of an entry here is just a tool for labelling the entries of H as to which term in the superposition $H = \sum_{m=1}^{(d+1)^2 n^6} H_m$ they sit in. Each term in the summation with have a unique colour. We will define the colour such that each superposition term has the properties required for Lemma 2.4 to hold.

$col_H(i,j) = col_H(j,i)$. We define H_m to contain all entries from H that are a particular colour m. As each entry of H_m has a single assigned colour, we can reconstruct H by a summation over all colours ($H = \sum H_m$). As H is Hermitian and each H_m is symmetric about the diagonal (from $col_H(i,j) = col_H(j,i)$), it follows that every H_m is also Hermitian. Also as H is row sparse and row computable, it follows that each H_m also has these properties. From this, one can see that the first requirement of Lemma 2.4 are satisfied.

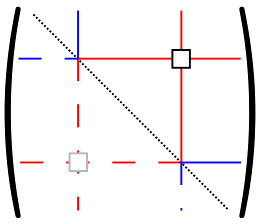

Figure 1: How the colouring of the matrix restricts the possible structure of H_m. See below for details.

From the restraint of $r_H(i,j)$ and $c_H(i,j)$ on the colour tuple, we know that for a non-zero element of a matrix H_m at position (i,j), there are no other non-zero elements in row i or column j (red lines in upper half of Figure 1). Furthermore, the other tuple colouring components $(k, i\text{mod}k, j\text{mod}k)$ create the case where there exists only a single non-zero element of H_m in the **row** j or **column** i in the position (j,i) (blue lines in upper half of Figure 1). This case is reflected for $i < j$ as $col_H(i,j) = col_H(j,i)$ (dashed lines in Figure 1). From these two conditions, it leads that each matrix H_m can be permuted to become a 2×2 block diagonal matrix where each mirror diagonal pair ($H_{i,j}$, $H_{j,i}$) forms a block. Therefore each H_m is a combinatorially block diagonal, proving the second part of Lemma 2.4. □

Lemma 2.5. (Simulation of 2×2 Block Matricies) *Every 2×2 combinatorially block diagonal, row-computable Hamiltonian is simulatable to within arbitrary polynomial approximation.*

In the interest of conciseness, the proof of Lemma 2.5 will not be given here but is available in Section 3.4.2 of [11]. Using Lemmas 2.4 and 2.5, it is now possible to show that a row-sparse, row-computable Hamiltonian on n qubits is simulatable.

Let H be a d-sparse Hamiltonian where $d \leq \text{poly}(\log N)$ and $\|H\| = \Lambda \leq \text{poly}(n)$. Our goal ($t > 0$) is to simulate e^{-itH} efficiently to within ϵ accuracy. We first express $H = \sum_{m=1}^{M} H_m$ from Lemma 2.4 where $M \leq (D+1)^2 n^6 \leq \text{poly}(n)$. Then using arguments from Lemma 2.2 we again arrive at Equation (11) with $l = (d+1)^2 n^6$. Lemma 2.5 shows us that each H_i can be efficiently simulated, and so we can approximately simulate e^{-iHt} to within the desired error. The complexity of this construction is given in Table 1 with respect to the number of queries that have to be made to the black-box defined in Definition 2.2.

2.3 Truncated Taylor Series - Berry *et al.* (2014)

We now discuss a method introduced by Berry *et al.* [12] for simulating finite-dimensional Hamiltonians of the form

$$H = \sum_{l=1}^{L} \alpha_l H_l, \tag{17}$$

where each H_l is unitary and implementable. Both local and sparse Hamiltonians can be decomposed into this form. The first step of the method is to break up the evolution of the unitary in Equation (3) into r segments, each of length t/r. The time evolution of each of these segments can be approximated as

$$U_r := e^{-iHt/r} \approx \sum_{k=0}^{K} \frac{(-iHt/r)^k}{k!}, \tag{18}$$

where the series has been truncated at order K. We know H is of the form in Equation (17), and so the truncated sum U_r can be expanded as

$$U_r \approx \sum_{k=0}^{K} \sum_{l_1,l_2,...,l_k=1}^{L} \frac{(-it/r)^k}{k!} \alpha_{l_1}\alpha_{l_2}...\alpha_{l_k} H_{l_1}H_{l_2}...H_{l_k}, \tag{19}$$

where each H_l is a unitary operator that is implementable and $\alpha_l > 0$. Note that the sum as a whole is not unitary due to truncation. This expression has the form

$$\tilde{U} = \sum_{j=0}^{m} \beta_j V_j, \tag{20}$$

where $V_j = (-i)^k H_{l_1}H_{l_2}...H_{l_k}$ and $\beta_j > 0$. This type of expression has be investigated previously by Kothari [13]. We now need to construct a mechanism for implementing $\tilde{U}$. If this sum were exactly unitary, one could use the oblvious amplitude amplification procedure described in [14]. However the sum is only close to unitary by an error that can be bounded (by choice of K) and so requires a variation of this technique. Here we first summarise the procedure in [14] and then show how this technique can be altered for non-unitary sums.

We have assumed that each unitary V_j is implementable, which can be abstracted as

$$S_V |j\rangle |\psi\rangle = |j\rangle V_j |\psi\rangle, \tag{21}$$

where $j \in 0, 1, ..., m$ and $|\psi\rangle$ is any arbitrary state. Before continuing, we define the unitary B to be

$$B|0\rangle = \frac{1}{\sqrt{s}} \sum_{j=0}^{m} \sqrt{\beta_j} |j\rangle, \tag{22}$$

where $s := \sum_{j=0}^{m} \beta_j$. By defining the operator

$$W := (B^\dagger \otimes \mathbb{I}) S_V (B \otimes \mathbb{I}), \tag{23}$$

one can show that

$$W |0\rangle |\psi\rangle = (B^\dagger \otimes \mathbb{I}) S_V (B \otimes \mathbb{I}) |0\rangle |\psi\rangle = \frac{1}{s} |0\rangle \tilde{U} |\psi\rangle + \sqrt{1 - \frac{1}{s^2}} |\Phi\rangle , \tag{24}$$

where $|\Phi\rangle$ is a state who's first qubit (the ancillary state) is supported in the subspace orthogonal to $|0\rangle$ [14]. Application of the projector $P := |0\rangle \langle 0| \otimes \mathbb{I}$ allows us to write

$$PW |0\rangle |\psi\rangle = \frac{1}{s} |0\rangle \tilde{U} |\psi\rangle . \tag{25}$$

The value of s can be varied by changing the number of segments r that we divide our evolution in to. For the oblivious amplitude amplification results of [14], we aim for $s = 2$. This then allows for the construction of $\tilde{U}$ by application of W, $W^\dagger$ and the reflection $R := \mathbb{I} - 2P$:

$$A |0\rangle |\psi\rangle = |0\rangle \tilde{U} |\psi\rangle , \tag{26}$$

where $A := -WRW^\dagger RW$. For the case of truncation however, we are dealing with a Taylor series that gives a good, but non-unitary, approximation $\tilde{U}$ of U_r. In this case the Equation (26) is not valid and so we require an alternative solution. By first observing that W is unitary, $P^2 = P$ and $P |0\rangle |\psi\rangle = |0\rangle |\psi\rangle$, one can see that

$$\begin{aligned} PA |0\rangle |\psi\rangle &= -PWRW^\dagger RW |0\rangle |\psi\rangle , \\ &= -PW(\mathbb{I} - 2P)W^\dagger (\mathbb{I} - 2P)W |0\rangle |\psi\rangle , \\ &= (3PW - 4PWPW^\dagger PW) |0\rangle |\psi\rangle . \end{aligned} \tag{27}$$

This equation can be further reduced by using $PWP = \frac{1}{s}(|0\rangle \langle 0| \otimes \tilde{U})$:

$$\begin{aligned} PA |0\rangle |\psi\rangle &= (3PW - 4PWPW^\dagger PW) |0\rangle |\psi\rangle , \\ &= (3PWP - 4PWPPW^\dagger PPWP) |0\rangle |\psi\rangle , \\ &= |0\rangle \left(\frac{3}{s} \tilde{U} - \frac{4}{s^3} \tilde{U} \tilde{U}^\dagger \tilde{U} \right) |\psi\rangle . \end{aligned} \tag{28}$$

This is a generalised form of the oblivious amplitude amplification result of Berry *et al.* [14]. If we add the restrictions that $|s - 2| = O(\delta)$ and $\|\tilde{U} - U_r\| = O(\delta)$, then we are close to being in the unitary regime and it follows that $\|\tilde{U}\tilde{U}^\dagger - \mathbb{I}\| = O(\delta)$ and

$$\| PA |0\rangle |\psi\rangle - |0\rangle U_r |\psi\rangle \| = O(\delta). \tag{29}$$

So if for each segment the error is $O(\delta)$, then if we take $\delta = O(\epsilon / r)$ we will be within the required total error.

The basis of this algorithm is the application of the unitary $A := -WRW^{\dagger}RW$ where $W := (B^{\dagger} \otimes \mathbb{I})S_V(B \otimes \mathbb{I})$. By construction, the application of A requires two instances of S_V, one instance of $S_V^{\dagger}$, and three instances of both B and $B^{\dagger}$ respectively. The complexity of each of these processes, along with other associated costs (like the number of ancillary states required) are considered in [12]. The result, for a Hamiltonian that is a sparse matrix with an oracle where the Hamiltonian is a sum of equal parts (α_l constant), is that the number of queries scales as

$$\tau \frac{\log(\tau/\epsilon)}{\text{loglog}(\tau/\epsilon)}, \tag{30}$$

where $\tau = d\|H\|_{max}t$.

2.4 Other Algorithms

There have been a number of alternative algorithms for the simulation of Hamiltonians since Lloyd in 1996. A brief introduction to some will be given in this section and the query complexity of each are given in Table 1. Childs in 2004 [15] gave a solution for reversible simulation of bipartite product Hamiltonians of the form $H = H_A \otimes H_B$. This was then followed by Berry *et al.* [10] who gave improvements on simulation of sparse Hamiltonians. Childs [16] was the first author to tackle the simulation of non-sparse Hamiltonians (he also performed analysis on sparse Hamiltonians), which was performed using discrete and continuous quantum walks. Childs later with Berry [17] produced another quantum walks algorithm which can be specifically applied to sparse Hamiltonian simulation. One of the more recent results is by Berry *et al.* [5] who have performed simulation of a sparse Hamiltonian via a combination of quantum walk methods and fractional-query simulation. They state that their simulation has near optimal dependence on all parameters (discussed below).

3 Discussion

There are a number of lower bounds for the complexity scaling of the simulation algorithms. Berry *et al.* [10] were the first to show that sublinear time scaling is not possible and so an algorithm must use $\Omega(t)$ queries. A sublinear scaling would be an algorithm that for sufficiently large t would grow slower in complexity than a linear function. This result is however only valid for the 'black box' setting, where the Hamiltonian H takes the form of a black box that can be queried. Improvements to this bound how been completed by Berry *et al.* [5] who produce a bound of $\Omega(\tau)$ where $\tau := d\|H\|_{max}t$. This means that the complexity must be at least linear in the product of the sparsity and the evolution time. Note that this bound is stronger than proof of $\Omega(t)$ and $\Omega(d)$ independently, since this would only provide a bound of $\Omega(t + d)$ which is weaker than the $\Omega(td)$ bound. This bound also proves that the dependence of [17] and [16] is optimal and that [5] is near optimal. A bound for the allowed error dependence ϵ has also been shown by Berry *et al.* to be $\Omega(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)})$. This proves optimal dependence on this parameter by [5], [14] and [12].

The query complexity of a number simulation algorithms are presented in Table 1. Quantum simulation algorithms are inherently difficult to compare as they often have very distinct constructions and are valid for particular situations. For Table 1, the "Query Complexity" signals either the number of operations that need to be performed or the number of queries that need to be sent to the Hamiltonian black box. For the black box algorithms, there is a secondary complexity which involves the number of additional 2-qubit gates that need to be performed along with the black box queries. This complexity will not be discussed here but is analysed in a number of different papers (see [14] for example).

Source	Query complexity
S.Lloyd (1996) [2]	$\text{poly}(\log N)(\|H\|t)^2/\epsilon$
Aharonov & Ta-Shma (2003) [11]	$\text{poly}(d, \log N)(\|H\|t)^{3/2}/\sqrt{\epsilon}$
Childs (2004) [15]	$(d^4 \log^4(N\|H\|t))^{(1+\delta)}/\epsilon^\delta$ (for any $\delta > 0$)
Berry *et al.* (2007) [10]	$(d^4 \log^*(N\|H\|t))^{(1+\delta)}/\epsilon^\delta$ (for any $\delta > 0$)
Childs & Kothari (2011) [18]	$(d^3 \log^*(N\|H\|t))^{(1+\delta)}/\epsilon^\delta$ (for any $\delta > 0$)
Childs (2010) [16], Berry & Childs (2012) [17]	$d\|H\|_{max}t/\sqrt{\epsilon}$
Berry *et al.* (2014) [12, 14]	$\tau\frac{\log(\tau/\epsilon)}{\log\log(\tau/\epsilon)}$ where $\tau = d^2\|H\|_{max}t$
Berry *et al.* (2015) [5]	$\tau\frac{\log(\tau/\epsilon)}{\log\log(\tau/\epsilon)}$ where $\tau = d\|H\|_{max}t$

Table 1: Comparison of simulation algorithms. The parameters are defined as follows: N dimensional matrix, d sparsity, t evolution time, ϵ allowed error. Here $\log^*(n) = min\{r|\log^{(r)}(n) < 2\}$ where ${}^{(r)}$ denotes an iterative logarithm ($r = 2 \Rightarrow \log(\log(n))$). The quantity $\|H\|_{max}$ denotes the largest entry of H in absolute value. Table originally presented by Childs [19].

We begin first by looking at how the latest result of Berry [5] varies with different scaling parameters (Figure 2). The graphs show how the query complexity varies with sparsity d, time t and error ϵ. The first observation is that the scaling in time appears to be very approximately linear, showing that the $\frac{log(\tau/\epsilon)}{loglog(\tau/\epsilon)}$ term offers a near constant contribution. This means that the algorithm is indeed close to the optimal linear time scaling. We can also see for the error scaling plot (right, Figure 2), that for a required error bound > 0.1, the query complexity is approximately constant. Although the position of this transition will vary with the input parameters, it is useful to know that there is approximately a threshold above which you will see very little gain in computational time for a larger error result.

We now look to compare how this algorithm compares with other results,

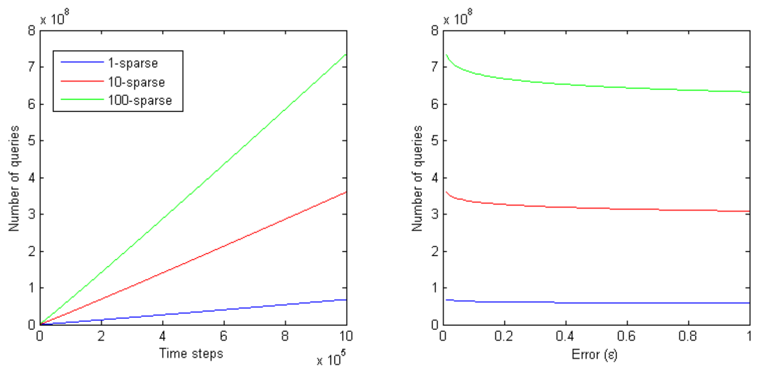

Figure 2: A plot showing the results of [5] ($O(\tau\frac{\log(\tau/\epsilon)}{\log\log(\tau/\epsilon)})$) where $\tau = d\|H\|_{max}t$. The left plot shows scaling with simulation time t for different sparsitys d. The error in this case was set to $\epsilon = 0.01$. The right plot shows scaling with error for a constant simulation time of $t = 10^6$. For each plot $\|H\|_{max} = 1$.

specifically that of [14, 12] and [16, 17] which have a complexity scaling of $O(\tau\frac{\log(\tau/\epsilon)}{\log\log(\tau/\epsilon)})$ and $O(d\|H\|_{max}t/\sqrt{\epsilon})$ respectively. Figure 3 compares how these algorithms scale with increasing simulation time for different values ϵ, where d and $\|H\|_{max}$ are constant. Conversely Figure 4 compares the scaling with respect to error, where now the constant terms are t, d and $\|H\|_{max}$.

We can see from the plots of Figure 3, the most efficient algorithm for a particular allowed error can vary depending on that parameters of the system. However we can see that for a particular situation, if you have an algorithm that is most efficient at a certain number of time steps, it is likely to still be the most efficient at all other time steps. This will not be true very large simulation times, as the results of Childs and Berry (2010/2012) [16, 17] have been shown to have optimal dependence on t, meaning that the will eventually become most efficient for any valid system parameters.

The plots of Figure 4 compare the complexity with increasing allowed error. What we can see is that for larger values of error, Childs/Berry 2010/2012[16, 17] outperforms more recent results. The results of 2014 and 2015 have approximately constant scaling with error, whereas 2010/2012 varies quite dramatically eventually becoming less efficient than both algorithms at low ϵ. The optimal scaling in t of 2010/2012 becomes apparent when looking at the value of ϵ where 2010/2012 and 2014 require the same number of queries. For example for $t = 10^6$, this point is at $\epsilon \sim 14 \times 10^{-5}$ (see right plot Figure 4) and for $t = 10^{12}$, $\epsilon \sim 8 \times 10^{-5}$.

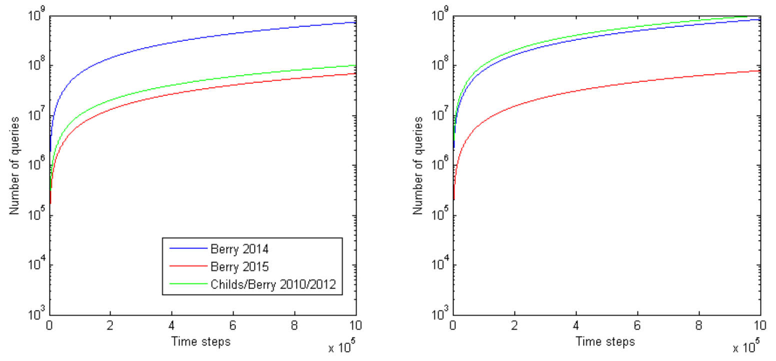

Figure 3: Algorithm scaling with increasing simulation time (time steps). The constants for the simulation were set at $\|H\|_{max} = 1$ and $d = 10$. The left image displays the results for $\epsilon = 0.01$ and the right for $\epsilon = 0.0001$. The legend is valid for both plots.

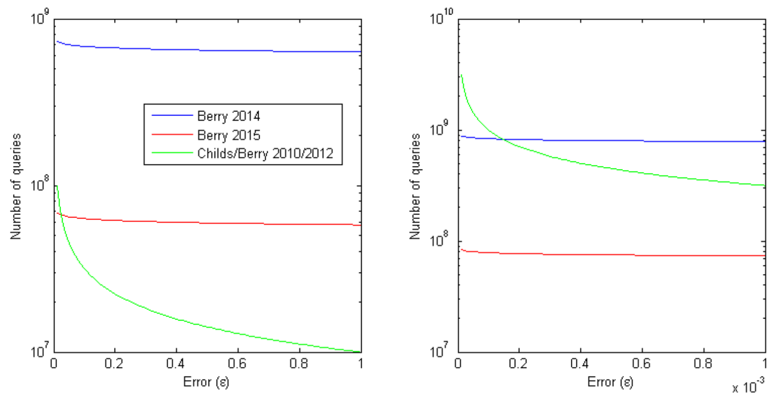

Figure 4: Algorithm scaling with increasing allowed error ϵ. The constants for the simulation were set at $\|H\|_{max} = 1$, $t = 10^6$ and $d = 10$. Both left and right images display the same function, but are plotted over different ranges of ϵ. The legend is valid for both plots.

4 Conclusion

We have introduced and discussed a wide range of algorithms within this essay. Following first from the early beginnings of Lloyd, Aharonov and Ta-Shma we ended with the current best known algorithms of Berry *et al.* and Childs. We have show the optimal bounds of these algorithms and that some algorithms are already at or close to these bounds. There is yet no algorithm that is both optimally bound in both t and ϵ scaling. As stated previously, comparison of algorithms is difficult to do in the general case. We have however outlined some characteristics of the algorithms which give an insight into their properties. The result of Berry *et al.* [5] is arguably the best algorithm we have currently, as it is optimal in ϵ, near optimal in t, and can support simulation of time-dependant sparse Hamiltonians (which covers a large number of applications). The result by Childs in contrast is optimal in t but only applies for time-independent Hamiltonians.

Childs has outlined a number of open questions and avenues for future work [7]. An algorithm with all optimal dependence would be advantageous, and particularly one that could apply to a wide range of Hamiltonian types (non-sparse, time dependant etc...). Following from this, it would be useful to start applying these algorithms to specific applications. This would require a better understanding of possible applications and systems that may require quantum simulation. A perhaps more distant step would be to implement some of these algorithms on some small scale systems.

References

[1] R. Feynman, "Simulating physics with computers," *International Journal of Theoretical Physics*, vol. 21, no. 6, p. 467, 1982.

[2] S. Lloyd, "Universal quantum simulators," *Science*, vol. 273, no. 5278, pp. 1073–1078, 1996.

[3] K. L. Brown, W. J. Munro, and V. M. Kendon, "Using quantum computers for quantum simulation," *Entropy*, vol. 12, no. 11, pp. 2268–2307, 2010.

[4] K. De Raedt, K. Michielsen, H. De Raedt, B. Trieu, G. Arnold, M. Richter, T. Lippert, H. Watanabe, and N. Ito, "Massively parallel quantum computer simulator," *Computer Physics Communications*, vol. 176, no. 2, pp. 121–136, 2007.

[5] D. W. Berry, A. M. Childs, and R. Kothari, "Hamiltonian simulation with nearly optimal dependence on all parameters," *arXiv preprint arXiv:1501.01715*, 2015.

[6] A. M. Childs and R. Kothari, "Limitations on the simulation of non-sparse hamiltonians," *arXiv preprint arXiv:0908.4398*, 2009.

[7] A. Childs, "Exponential improvement in precision for simulating sparse hamiltonians," A Publisher, 2014.

[8] C. M. Dawson and M. A. Nielsen, "The Solovay-Kitaev algorithm," *arxiv*, 2005.

[9] M. Suzuki, "General theory of higher-order decomposition of exponential operators and symplectic integrators," *Physics Letters A*, vol. 165, no. 5, pp. 387–395, 1992.

[10] D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders, "Efficient quantum algorithms for simulating sparse hamiltonians," *Communications in Mathematical Physics*, vol. 270, no. 2, pp. 359–371, 2007.

[11] D. Aharonov and A. Ta-Shma, "Adiabatic quantum state generation and statistical zero knowledge," in *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pp. 20–29, ACM, 2003.

[12] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, and R. D. Somma, "Simulating hamiltonian dynamics with a truncated taylor series," *arXiv preprint arXiv:1412.4687*, 2014.

[13] R. Kothari, *Efficient algorithms in quantum query complexity.* PhD thesis, University of Waterloo, Ontario, 2014.

[14] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, and R. D. Somma, "Exponential improvement in precision for simulating sparse hamiltonians," in *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pp. 283–292, ACM, 2014.

[15] A. M. Childs, D. W. Leung, and G. Vidal, "Reversible simulation

of bipartite product hamiltonians," *Information Theory, IEEE Transactions on*, vol. 50, no. 6, pp. 1189–1197, 2004.

[16] A. M. Childs, "On the relationship between continuous- and discrete-time quantum walk," *Communications in Mathematical Physics*, vol. 294, no. 2, pp. 581–603, 2010.

[17] D. W. Berry and A. M. Childs, "Black-box hamiltonian simulation and unitary implementation," *Quantum Info. Comput.*, vol. 12, pp. 29–62, Jan. 2012.

[18] A. M. Childs and R. Kothari, "Simulating sparse hamiltonians with star decompositions," in *Theory of Quantum Computation, Communication, and Cryptography*, pp. 94–103, Springer, 2011.

[19] A. M. Childs, "Exponential improvement in precision for simulating sparse hamiltonians." http://www.nist.gov/itl/math/upload/slides_andrew_childs.pdf, 2014. Presentation given at the National Institute of Standards and Technology.