Efficient quantum algorithms for simulating sparse Hamiltonians

Dominic W. Berry,[1,2] Graeme Ahokas,[2,3] Richard Cleve,[2,3,4,5] and Barry C. Sanders[2,6]

[1] *Department of Physics, The University of Queensland, Queensland 4072, Australia*
[2] *Institute for Quantum Information Science, University of Calgary, Alberta T2N 1N4, Canada*
[3] *Department of Computer Science, University of Calgary, Alberta T2N 1N4, Canada*
[4] *School of Computer Science, University of Waterloo, Ontario N2L 3G1, Canada*
[5] *Institute for Quantum Computing, University of Waterloo, Ontario N2L 3G1, Canada*
[6] *Centre for Quantum Computer Technology, Macquarie University, Sydney, New South Wales 2109, Australia*

We present an efficient quantum algorithm for simulating the evolution of a sparse Hamiltonian H for a given time t in terms of a procedure for computing the matrix entries of H. In particular, when H acts on n qubits, has at most a constant number of nonzero entries in each row/column, and $\|H\|$ is bounded by a constant, we may select any positive integer k such that the simulation requires $O((\log^* n)t^{1+1/2k})$ accesses to matrix entries of H. We show that the temporal scaling cannot be significantly improved beyond this, because sublinear time scaling is not possible.

I. INTRODUCTION

There are three main applications of quantum computer algorithms: the hidden subgroup problem, with Shor's factorization algorithm one important example [1], search problems [2], and simulation of quantum systems [3, 4]. Lloyd's method for simulating quantum systems [4] assumes a tensor product structure of smaller subsystems. Aharonov and Ta-Shma (ATS) [5] consider the alternative case where there is no evident tensor product structure to the Hamiltonian, but it is sparse and there is an efficient method of calculating the nonzero entries in a given column of the Hamiltonian. Such respresentations of Hamiltonians can arise as encodings of computational problems, such as simulations of quantum walks [6, 7, 8, 9, 10] or adiabatic quantum computations [11].

Here we apply the higher-order integrators of Suzuki [12, 13] to reduce the temporal scaling from $t^{3/2}$ [5] or t^2 [4] to the slightly superlinear scaling $t^{1+1/2k}$, where k is the order of the integrator and may be an arbitrarily large integer. We determine an upper bound on the number of exponentials required to approximate the evolution with a given accuracy. This enables us to estimate the optimal value of k, and therefore the k-independent scaling in t. We than prove that, in the black-box setting, this scaling is close to optimal, because it is not possible to perform simulations sublinear in t. We also provide a superior method for decomposing the Hamiltonian into a sum for the problem considered by ATS, which dramatically reduces the scaling of n^2 [14] or n^9 [5] to $\log^* n$ for n qubits. This method is similar to "deterministic coin tossing" [15], as well as Linial's graph coloring method [16].

II. PROBLEMS AND RESULTS

We commence with a statement of the problems that we consider in this paper and follow with the solutions that will be proven.

Problem 1. *The Hamiltonian is of the form $H = \sum_{j=1}^{m} H_j$. The problem is to simulate the evolution e^{-iHt} by a sequence of exponentials $e^{-iH_j t'}$ such that the maximum error in the final state, as quantified by the trace distance, does not exceed ϵ. Specifically we wish to determine an upper bound on the number of exponentials, $N_{\exp}$, required in this sequence.*

For this problem, the H_j should be of a form that permits $e^{-iH_j t'}$ to be accurately and efficiently simulated for arbitrary evolution time t'. It is therefore reasonable to quantify the complexity of the calculation by the number of exponentials required. This problem includes the physically important case of simulating tensor product systems considered by Lloyd [4], for which each H_j can be considered to be an interaction Hamiltonian. It also may be applied to the case where there is a procedure for calculating the nonzero elements in the columns [5]. In that case, each H_j is a 1-sparse Hamiltonian. The decomposition must be calculated, which requires additional steps in the algorithm.

Our general result for Problem 1 is the following theorem.

Theorem 1. *When the permissible error is bounded by ϵ, $N_{\exp}$ is bounded by*

$$N_{\exp} \leq 2m5^{2k}(m\tau)^{1+1/2k}/\epsilon^{1/2k}, \tag{1}$$

for $\epsilon \leq 1 \leq 2m5^{k-1}\tau$, where $\tau = \|H\|t$, and k is an arbitrary positive integer.

By taking k to be sufficiently large, it is possible to obtain scaling that is arbitrarily close to linear in τ. However, for a given value of τ, taking k to be too large will increase $N_{\rm exp}$. To estimate the optimum value of k to take, we express Eq. (1) as

$$N_{\rm exp} \leq 2m^2\tau\, e^{2k\ln 5+\ln(m\tau/\epsilon)/2k}.$$

The right-hand side has a minimum for

$$k = \text{round}\left[\frac{1}{2}\sqrt{\log_5(m\tau/\epsilon)+1}\right].$$

Here we have added 1 and rounded because k must take integer values. Adopting this value of k provides the upper bound

$$N_{\rm exp} \leq 4m^2\tau\, e^{2\sqrt{\ln 5\ln(m\tau/\epsilon)}}, \tag{2}$$

for $\epsilon \leq 1 \leq m\tau/25$. Eq. (2) is an expression for $N_{\rm exp}$ that is independent of k.

The scaling in Eq. (2) is close to linear for large $m\tau$. We show that this scaling is effectively optimal, because it is not possible to perform general simulations sublinear in τ (see Sec. IV). This result applies in the "black-box" setting, so it does not rule out the possibility that individual Hamiltonians have structure which allows them to be simulated more efficiently.

The second problem which we consider is that of sparse Hamiltonians.

Problem 2. *The Hamiltonian H has no more than d nonzero entries in each column, and there exists a black-box function f that gives these entries. The dimension of the space which H acts upon does not exceed 2^n. If the nonzero elements in column x are $y_1,\ldots,y_{d'}$, where $d' \leq d$, then $f(x,i) = (y_i, H_{x,y_i})$ for $i \leq d'$, and $f(x,i) = (x,0)$ for $i > d'$. The problem is to simulate the evolution e^{-iHt} such that the maximum error in the final state, as quantified by the trace distance, does not exceed ϵ. We wish to determine the scaling of the number of calls to f, $N_{\rm bb}$, required for this simulation.*

For each x, the order in which the corresponding y_i are given can be arbitrary. The function f is an arbitrary black-box function, but we assume that there is a corresponding unitary U_f such that

$$U_f|x,i\rangle|0\rangle = |\phi_{x,i}\rangle|y_i, H_{x,y_i}\rangle,$$

and we may perform calls to both U_f and $U_f^\dagger$. Here $|\phi_{x,i}\rangle$ represents any additional states which are produced in the reversible calculation of f.

ATS approached the problem by decomposing the Hamiltonian into a sum of H_j. We apply a similar approach to obtain the following theorem.

Theorem 2. *The number of black-box calls for given k is*

$$N_{\rm bb} \in O\left((\log^* n)d^2 5^{2k}(d^2\tau)^{1+1/2k}/\epsilon^{1/2k}\right) \tag{3}$$

with $\log^* n \equiv \min\{r|\log_2^{(r)} n < 2\}$ (the $^{(r)}$ indicating the iterated logarithm).

The $\log^* n$ scaling is a dramatic improvement over the n^9 scaling implicit in the method of ATS, as well as the n^2 scaling of Childs [14].

III. HIGHER ORDER INTEGRATORS

To prove Theorem 1, we apply the method of higher-order integrators. Following Suzuki [12, 13], we define

$$S_2(\lambda) = \prod_{j=1}^{m} e^{H_j\lambda/2} \prod_{j'=m}^{1} e^{H_{j'}\lambda/2},$$

which is the basic Lie-Trotter product formula, and the recursion relation

$$S_{2k}(\lambda) = [S_{2k-2}(p_k\lambda)]^2 S_{2k-2}((1-4p_k)\lambda)[S_{2k-2}(p_k\lambda)]^2$$

with $p_k = (4 - 4^{1/(2k-1)})^{-1}$ for $k > 1$. Suzuki then proves that [12]

$$\left\| \exp\left(\sum_{j=1}^{m} H_j \lambda \right) - S_{2k}(\lambda) \right\| \in O(|\lambda|^{2k+1}) \tag{4}$$

for $|\lambda| \to 0$. The parameter λ corresponds to $-it$ for Hamiltonian evolution.

We first assess the higher-order integrator method in terms of all quantities t, m, k, and $\|H\|$. Our result is

Lemma 1. *Using integrators of order k and dividing the time into r intervals, we have the bound*

$$\left\| \exp\left(-it \sum_{j=1}^{m} H_j \right) - [S_{2k}(-it/r)]^r \right\| \leq 5(2 \times 5^{k-1} m\tau)^{2k+1}/r^{2k}, \tag{5}$$

for

$$\begin{aligned} 4m5^{k-1}\tau/r &\leq 1, \\ (16/3)(2 \times 5^{k-1} m\tau)^{2k+1}/r^{2k} &\leq 1. \end{aligned} \tag{6}$$

Proof. Consider a Taylor expansion of both terms in the left-hand side (LHS) of Eq. (4). Those terms containing λ to powers less than $2k+1$ must cancel because the correction term is $O(|\lambda|^{2k+1})$, and terms with $\lambda^{2k'+1}$ for $k' \geq k$ must contain a product of $2k'+1$ of the H_j terms; thus

$$\exp\left(\sum_{j=1}^{m} H_j \lambda \right) = S_{2k}(\lambda) + \sum_{k'=k}^{\infty} \lambda^{2k'+1} \sum_{l=1}^{L_{k'}} C_l^{k'} \prod_{q=1}^{2k'+1} H_{j_{lq}}.$$

The constants $C_l^{k'}$ and the number of terms $L_{k'}$ depend on m and k.

In order to bound $C_l^{k'}$ and $L_{k'}$, first consider the Taylor expansion of the exponential in the LHS of Eq. (4). Because the operators H_j are in general noncommuting, expanding $(H_1 + \cdots + H_m)^{2k'+1}$ yields $m^{2k'+1}$ terms. Therefore the Taylor expansion contains $m^{2k'+1}$ terms with $\lambda^{2k'+1}$. These terms have multiplying factors of $1/(2k'+1)!$ because this is the multiplying factor given by the Taylor expansion of the exponential.

To place a bound on the number of terms in the Taylor expansion of $S_{2k}(\lambda)$, note that $S_{2k}(\lambda)$ consists of a product of

$$2(m-1)5^{k-1} + 1$$

exponentials. The expansion for $S_{2k}(t)$ may be obtained by expanding each of the exponentials individually. There will be no more than

$$[2(m-1)5^{k-1} + 1]^{2k'+1}$$

terms with $\lambda^{2k'+1}$. Because $|p_k| < 1$ and $|1 - 4p_k| < 1$, the multiplying factors for each of these terms must be less than 1.

Each H_j satisfies $\|H_j\| \leq \|H\|$ [5]. Defining $\Lambda \equiv \|H\|$, and using standard inequalities we obtain

$$\begin{aligned} &\left\| \sum_{k'=k}^{\infty} \lambda^{2k'+1} \sum_{l=1}^{L_{k'}} C_l^{k'} \prod_{q=1}^{2k'+1} H_{j_{lq}} \right\| \leq \sum_{k'=k}^{\infty} |\lambda\Lambda|^{2k'+1} L_{k'} \\ &\leq \sum_{k'=k}^{\infty} |\lambda\Lambda|^{2k'+1} \left\{ m^{2k'+1} + [2(m-1)5^{k-1} + 1]^{2k'+1} \right\} \\ &\leq 2 \sum_{k'=k}^{\infty} |\lambda\Lambda|^{2k'+1} [2m5^{k-1}]^{2k'+1} = \frac{2|2m5^{k-1}\lambda\Lambda|^{2k+1}}{1 - |2m5^{k-1}\lambda\Lambda|^2}. \end{aligned}$$

Therefore we obtain the inequality

$$\left\| \exp\left(\lambda \sum_{j=1}^{m} H_j \right) - S_{2k}(\lambda) \right\| \leq (8/3)|2m5^{k-1}\Lambda\lambda|^{2k+1},$$

provided $|2m5^{k-1}\Lambda\lambda| \leq 1/2$. Substituting $\lambda = -it/r$ where r is an integer, and taking the power of r, gives the error bound

$$\left\| \exp\left(-it\sum_{j=1}^{m} H_j\right) - [S_{2k}(-it/r)]^r \right\| \leq [1 + (8/3)(2m5^{k-1}\Lambda t/r)^{2k+1}]^r - 1, \tag{7}$$

for $4m5^{k-1}\Lambda t/r \leq 1$. This may alternatively be expressed as in Lemma 1. □

By placing limits on the norm of the difference in the unitaries, we limit the trace distance of the output states. This is because

$$\begin{aligned} \|U_1 - U_2\| &\geq \|U_1|\psi\rangle - U_2|\psi\rangle\| \\ &\geq \frac{1}{2}\mathrm{Tr}\left|U_1|\psi\rangle\langle\psi|U_1^\dagger - U_2|\psi\rangle\langle\psi|U_2^\dagger\right| \\ &= D\left(U_1|\psi\rangle\langle\psi|U_1^\dagger, U_2|\psi\rangle\langle\psi|U_2\right), \end{aligned}$$

with D the trace distance. We now use this to prove Theorem 1.

Proof. (of Theorem 1) Let us take

$$r = \lceil 4 \times 5^{k-1/2}(m\tau)^{1+1/2k}/\epsilon^{1/2k} \rceil. \tag{8}$$

Given the restrictions $\epsilon \leq 1 \leq 2m5^{k-1}\tau$, it is easily seen that Eqs. (6) hold. In addition, the right-hand side of Eq. (5) does not exceed ϵ, so the error can not exceed ϵ.

Because the number of exponentials in $S_{2k}(\lambda)$ does not exceed $2m5^{k-1}$, we have $N_{\exp} \leq 2m5^{k-1}r$. If we take r as in Eq. (8), then we find that

$$N_{\exp} \leq 2m5^{2k}(m\tau)^{1+1/2k}/\epsilon^{1/2k}. \tag{9}$$

Here the multiplying factor has been changed to take into account the ceiling function. Hence the order scaling is as in Eq. (1). □

This result may be used for any case where the Hamiltonian is a sum of terms that may be simulated efficiently. It may therefore be applied to the case of tensor product systems, where the individual H_j are interaction Hamiltonians. It can be also used for cases of the type of Problem 2, where the Hamiltonian is sparse. In this case we have the additional task of decomposing the Hamiltonian into a sum.

IV. LINEAR LIMIT ON SIMULATION TIME

We have shown that the simulation of any Hamiltonian may be performed arbitrarily close to linearly in the scaled time τ. We now show that the scaling cannot be sublinear in τ, provided the number of qubits can grow at least logarithmically with respect to τ. The result is

Theorem 3. *For all positive integers N there exists a row-computable 2-sparse Hamiltonian H such that simulating the evolution of H for scaled time $\tau = \pi N/2$ within precision $1/4$ requires at least $\tau/2\pi$ queries to H.*

Here a row-computable Hamiltonian means one where there is a method for efficiently calculating the nonzero elements in each row.

Proof. The idea is to construct a 2-sparse Hamiltonian such that the simulation of this Hamiltonian determines the parity of N bits. It has been shown that the parity of N bits requires $N/2$ queries to compute within error $1/4$ [19]; therefore the Hamiltonian can not be simulated any more efficiently.

First consider a Hamiltonian H acting on orthogonal basis states $|0\rangle, \ldots, |N\rangle$, for which the nonzero matrix entries are

$$\langle j+1|H|j\rangle = \langle j|H|j+1\rangle = \sqrt{(N-j)(j+1)}/2.$$

This Hamiltonian is equivalent to a J_x operator for a spin $N/2$ system, with the $|j\rangle$ being J_z eigenstates. It is therefore clear that $e^{-i\pi H}|0\rangle = |N\rangle$ and $\|H\| = N/2$.

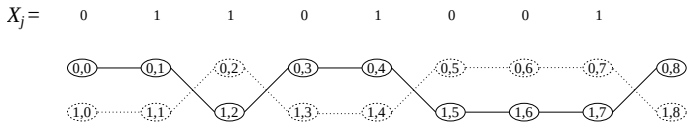

FIG. 1: Graph representing example the Hamiltonian in the proof of Theorem 3. States are represented by ellipses, and nonzero elements of the Hamiltonian are indicated by lines. The sequence of states $|k_j, j\rangle$ with $k_0 = 0$ is indicated by the solid line.

Now we construct an augmented version of the above Hamiltonian, that corresponds to a graph with two disjoint lines with weights as above, where the lines "cross over" at the positions where bits $X_1, \ldots, X_N$ are 1. We add an ancilla qubit so the Hamiltonian H acts on basis states $|0,0\rangle, \ldots, |0,N\rangle$, $|1,0\rangle, \ldots, |1,N\rangle$. The nonzero matrix entries of H are

$$\langle k', j+1|H|k,j\rangle = \langle k,j|H|k',j+1\rangle = \sqrt{(N-j)(j+1)}/2$$

for values of k and k' such that $k \oplus k' = X_{j+1}$ (where $\oplus$ is XOR).

Thus, if X_{j+1} is zero, then there is a nonzero matrix element between $|0,j\rangle$ and $|0,j+1\rangle$, as well as between $|1,j\rangle$ and $|1,j+1\rangle$. If X_{j+1} is equal to 1, then the nonzero matrix elements are between $|0,j\rangle$ and $|1,j+1\rangle$, as well as $|1,j\rangle$ and $|0,j+1\rangle$. We may determine a sequence of bits $k_0, \ldots, k_N$ such that $k_j \oplus k_{j+1} = X_{j+1}$. The Hamiltonian acting on the set of states $|k_j, j\rangle$ will then be identical to that acting on the states $|j\rangle$ with the original Hamiltonian. It is therefore clear that $e^{-i\pi H}|k_0, 0\rangle = |k_N, N\rangle$.

The graph corresponding to a Hamiltonian of this type is shown in Fig. 1. The system separates into two distinct sets of states which are not connected. If the system starts in one of the states on the path indicated by the solid line, it can not evolve under the Hamiltonian to a state on the dotted line. From the definition of the k_j, if $k_0 = 0$, then k_j is the parity of bits X_1 to X_j, and in particular k_N gives the parity of all N bits. Thus if we start with the initial state $|0,0\rangle$ and simulate the evolution $e^{-i\pi H}$, we obtain the state $|k_N, N\rangle$, where k_N is the parity of the N bits $X_1, \ldots, X_N$. Thus measuring the state of the ancilla qubit will give the parity.

Let us denote the final state obtained by the simulation by $|\psi\rangle$, and the reduced density operator for the ancilla by $\rho_{\rm anc}$. If the error probability is no less than 1/4, then $D(\rho_{\rm anc}, |k_N\rangle\langle k_N|) \geq 1/4$, which implies that

$$D(|\psi\rangle\langle\psi|, |k_N, N\rangle\langle k_N, N|) \geq 1/4.$$

Hence, if there are no more than $N/2$ queries to the X_j, the error in the simulation as quantified by the trace distance must be at least 1/4.

Each query to a column of H requires no more than two queries to the X_j (for column j we require a query to X_j and X_{j+1}). Thus, if there are no more than $N/4$ queries to H, then there are no more than $N/2$ queries to the X_j. In addition, the scaled time for the simulation is

$$\tau = \|H\|t = \pi N/2.$$

Thus the simulation of H requires at least $N/4 = \tau/2\pi$ queries to obtain trace distance error no more than 1/4. □

The form of this result differs slightly from that in the previous section, in that the cost is specified in terms of the number of queries to the Hamiltonian, rather than the number of exponentials. It is straightforward to show the following result for the number of exponentials.

Corollary 1. *There is no general integrator for Hamiltonians of the form $H = H_1 + H_2$ such that (trace distance) error $\leq 1/4$ may be achieved with the number of exponentials $N_{\rm exp} < \tau/2\pi$.*

By general integrator we mean an integrator that depends only on τ, and not the Hamiltonian.

Proof. We take H as in the preceding proof. This Hamiltonian may be expressed in the form $H = H_1 + H_2$ by taking H_1 to be the Hamiltonian with $\langle k', j+1|H_1|k,j\rangle$ nonzero only for even j, and H_2 to be the Hamiltonian with $\langle k', j+1|H_2|k,j\rangle$ nonzero only for odd j. Each query to the H_k requires only one query to the X_j. For H_1 (H_2), determining the nonzero element in column j requires determining which of j and $j+1$ is odd (even), and performing a query to the corresponding X_j or X_{j+1}.

Both H_1 and H_2 are 1-sparse, and therefore may be efficiently simulated with only two queries [17, 18]. If $N_{\rm exp} < \tau/2\pi$, the total number of queries to the X_j is no more than τ/π. Taking $t = \pi$ and $\|H\| = N/2$, the number of queries is no more than $N/2$. However, from Ref. [19] the error rate can be no less than 1/4. Hence error rate $\leq 1/4$ can not be achieved with $N_{\rm exp} < \tau/2\pi$. □

V. EFFICIENT DECOMPOSITION OF HAMILTONIAN

Next we consider the problem of simulating general sparse Hamiltonians, as in Problem 2. Given that the dimension of the space does not exceed 2^n, we may represent the state of the system on n qubits, and x and y may be n-bit integers. The real and imaginary parts of the matrix elements will be represented by n' bit integers (for a total of $2n'$ bits for each matrix element), where n' must be chosen large enough to achieve the desired accuracy.

In order to simulate the Hamiltonian, we decompose it into the form $H = \sum_{j=1}^{m} H_j$, where each H_j is *1-sparse* (i.e., has at most one nonzero entry in each row/column). If H_j is 1-sparse then it is possible to directly simulate $\exp(-iH_j t)$ with just two black-box queries to H_j [17, 18]. Since the value of m directly impacts the total cost of simulating H, it is desirable to make m as small as possible. The size of the sum may be limited as in the following lemma.

Lemma 2. *There exists a decomposition $H = \sum_{j=1}^{m} H_j$, where each H_j is 1-sparse, such that $m = 6d^2$ and each query to any H_j can be simulated by making $O(\log^* n)$ queries to H.*

Proof. From the black-box function for H, we wish to determine black-box functions for each H_j that give the nonzero row number, y, and matrix element corresponding to each column x. This black-box for H_j is represented by the function $g(x, j)$, with output $(y, (H_j)_{x,y})$. If there is no nonzero element in column x, the output is $(x, 0)$.

Intuitively, it is helpful to consider the graph G_H associated with H whose vertex set is $\{0,1\}^n$. Each vertex corresponds to a row or column number, and there is an edge between vertex x and y if the matrix element $H_{x,y}$ is nonzero. As H is Hermitian we take the graph to be undirected. We wish to determine an "edge-coloring" of G_H, which is a labeling of the edges such that incident edges have different colors. Each edge color, j, then corresponds to a different Hamiltonian H_j in the decomposition of H.

The basic idea is as in the following labeling scheme, where the labels are indexed from the set $\{1, \ldots, d\}^2$. We take f_y to be the y-component of f; then $f_y(x, i)$ gives the i^{th} neighbor of vertex x in the graph. Let (x, y) be an edge of G_H such that $y = f_y(x, i)$ and $x = f_y(y, j)$. Thus edge (x, y) is labeled with the ordered pair (i, j) for $x \le y$, or (j, i) for $x \ge y$. This labeling is not quite an edge-coloring; for $w < x < y$ it is possible for edges (w, x) and (x, y) to both have the label (i, j). That will be the case if y and w are the i^{th} and j^{th} neighbors of x, respectively, and x is the i^{th} neighbor of w and the j^{th} neighbor of y. To ensure that the labels are unique, we add the additional parameter ν, so the label is (i, j, ν).

We assign ν via a method similar to "deterministic coin tossing" [15]. We set $x_0^{(0)} = x$, then determine a sequence of vertices

$$x_0^{(0)} < x_1^{(0)} < x_2^{(0)} < \cdots$$

such that $x_{l+1}^{(0)} = f_y(x_l^{(0)}, i)$ and $f_y(x_{l+1}^{(0)}, j) = x_l^{(0)}$. That is, the edges $(x_l^{(0)}, x_{l+1}^{(0)})$ are labeled (i, j, ν), with the same values of i and j for each edge. We need to choose values of ν for the edges such that the same value is never repeated in this chain.

A typical chain may have only two elements; however, there exist Hamiltonians such that long chains may be formed. In the case that the chain is long, we do not determine it any further than $x_{z_n+1}^{(0)}$. Here z_n is the number of times we must iterate $l \mapsto 2\lceil \log_2 l \rceil$ (starting at 2^n) to obtain 6 or less. This quantity is of order $\log^* n$, and for any realistic problem size z_n itself will be no more than 6 [21].

Now we determine a second sequence of values $x_l^{(1)}$. This sequence is taken to have the same length as the first sequence. For each $x_l^{(0)}$ and $x_{l+1}^{(0)}$, we determine the first bit position where these two numbers differ, and record the value of this bit for $x_l^{(0)}$, followed by the binary representation of this position, as $x_l^{(1)}$. The bit positions are numbered from zero; that is, the first bit is numbered $00\ldots0$. If $x_l^{(0)}$ is at the end of the sequence, we simply take $x_l^{(1)}$ to be the first bit of $x_l^{(0)}$, followed by the binary representation of 0. There are 2^n different possible values for each of the $x_l^{(0)}$, and $2n$ different possible values for each of the $x_l^{(1)}$.

From the definition, each $x_l^{(0)}$ is unique. Also $x_l^{(1)}$ must differ from $x_{l+1}^{(1)}$. This is because, even if the positions of the first bit where $x_l^{(0)}$ differs from $x_{l+1}^{(0)}$ and $x_{l+1}^{(0)}$ differs from $x_{l+2}^{(0)}$ are identical, the value of this bit for $x_l^{(0)}$ will of course be different from the value for $x_{l+1}^{(0)}$. As the $x_l^{(1)}$ contain both the position and the value of the bit, $x_l^{(1)}$ must differ from $x_{l+1}^{(1)}$.

There is a subtlety when $x_{l+1}^{(0)}$ is at the end of the sequence. Then $x_{l+1}^{(1)}$ contains the first bit of $x_{l+1}^{(0)}$, and the position of the first bit which differs is taken to be 1. In that case, if $x_l^{(0)}$ differs from $x_{l+1}^{(0)}$ at the first bit (so the bit

TABLE I: Example values of $x_l^{(p)}$ under our scheme for calculating ν. The value of ν obtained is in the upper right, and is shown in bold. For this example $n = 18$ and $z_n = 4$. The values in italics are those that may differ from $w_{l+1}^{(p)}$ (there are no corresponding values for the bottom row).

$l\backslash p$	0	1	2	3	4
0	001011100110011010	000001	0100	000	**000**
1	010110101010011011	000010	1100	100	*100*
2	011011101110101101	000000	0001	*000*	*000*
3	101011101011110100	010001	*1001*	*100*	*100*
4	101011101011110101	*000001*	*0000*	*000*	*000*
5	111000010110011010	100000	1000	100	100

positions recorded in $x_l^{(1)}$ and $x_{l+1}^{(1)}$ are identical), then the bit values which are recorded in $x_l^{(1)}$ and $x_{l+1}^{(1)}$ must be different. Thus it is still not possible for $x_l^{(1)}$ to be equal to $x_{l+1}^{(1)}$.

We repeat this process until we determine the sequence of values $x_l^{(z_n)}$. We determine the $x_l^{(p+1)}$ from the $x_l^{(p)}$ in exactly the same way as above. At each step, $x_l^{(p)}$ differs from $x_{l+1}^{(p)}$ for exactly the same reasons as for $p = 1$. As we go from p to $p + 1$, the number of possible values for the $x_l^{(p)}$ is reduced via the mapping $k \mapsto 2\lceil \log_2 k \rceil$. Due to our choice of z_n, there are six possible values for $x_0^{(z_n)}$.

Now if $w < x$ with $x = f_y(w, i)$ and $w = f_y(x, j)$, then we may set $w_0^{(0)} = w$ and perform the calculation in exactly the same way as for x in order to determine $w_0^{(z_n)}$. If the chain of $x_l^{(0)}$ ends before z_n, then the $x_l^{(p)}$ will be the same as the $w_{l+1}^{(p)}$. In particular $x_0^{(z_n)}$ will be equal to $w_1^{(z_n)}$, so it is clear that $w_0^{(z_n)}$ will differ from $x_0^{(z_n)}$.

On the other hand, if there is a full chain of $x_0^{(0)}$ up to $x_{z_n+1}^{(0)}$, then the chain for w will end at $w_{z_n+1}^{(0)}$, which is equivalent to $x_{z_n}^{(0)}$. Then $w_{z_n+1}^{(1)}$ will be calculated in a different way to $x_{z_n}^{(1)}$, and may differ. However, $w_{z_n}^{(1)}$ will be equal to $x_{z_n-1}^{(1)}$. At step p, $w_{z_n-p+1}^{(p)}$ will be equal to $x_{z_n-p}^{(p)}$. In particular, at the last step, $w_1^{(z_n)}$ will be equal to $x_0^{(z_n)}$. Thus we find that $w_0^{(z_n)}$ again differs from $x_0^{(z_n)}$.

As $x_0^{(z_n)}$ has this useful property, we assign the edge (x, y) the color (i, j, ν), where $\nu = x_0^{(z_n)}$. Due to the properties of the above scheme, if the edge (w, x) has the same values of i and j as (x, y), it must have a different value of ν. Therefore, via this scheme, adjacent edges must have different colors.

Now we describe how to calculate the black-box function g using this approach. We replace j with (i, j, ν) to reflect the labeling scheme, so the individual Hamiltonians are $H_{(i,j,\nu)}$. The black-box function we wish to calculate is $g(x, i, j, \nu)$. We also define the function $\Upsilon(x, i, j)$ to be equal to the index ν as calculated in the above way. There are three main cases where we give a nontrivial output:

1. $f_y(x, i) = x$, $i = j$ and $\nu = 0$,
2. $f_y(x, i) > x$, $f_y(f_y(x, i), j) = x$ and $\Upsilon(x, i, j) = \nu$,
3. $f_y(x, j) < x$, $f_y(f_y(x, j), i) = x$ and $\Upsilon(f_y(x, j), i, j) = \nu$.

In cases 1 and 2 we return $g(x, i, j, \nu) = f(x, i)$, and for case 3 we return $g(x, i, j, \nu) = f(x, j)$; in all other cases we return $g(x, i, j, \nu) = (x, 0)$.

Case 1 corresponds to diagonal elements of the Hamiltonian. We only return a nonzero result for $\nu = 0$, in order to prevent this element being repeated in different Hamiltonians $H_{(i,j,\nu)}$. Case 2 corresponds to there existing a $y > x$ such that y is the i^{th} neighbor of x and x is the j^{th} neighbor of y. Similarly Case 3 corresponds to there existing a $w < x$ such that w is the j^{th} neighbor of x and x is the i^{th} neighbor of w. The uniqueness of the labeling ensures that cases 2 and 3 are mutually exclusive.

As there are d possible values for i and j, and ν may take six values, there are $6d^2$ colors. Thus we may take $m = 6d^2$. In determining ν, we need a maximum of $2(z_n + 2)$ queries to the black-box; this is of order $\log^* n$. □

To illustrate the method for determining ν, an example is given in Table I for $\{x_l^{(p)}\}$ and Table II for $\{w_l^{(p)}\}$. Fig. 2 shows a portion of the graph corresponding to the values in Tables I and II. In the Tables $n = 18$, so there are 2^{18} possible values in the first column. Then there are 36 possible values in the second column, 12 in the third, 8 in the fourth and 6 in the fifth. Thus z_n is equal to 4 in this case, and the sequence of $x_l^{(0)}$ is determined up to $x_5^{(0)}$.

TABLE II: Example values of $w_l^{(p)}$ under our scheme for calculating ν. The value of ν obtained is in the upper right, and is shown in bold. For this example $n = 18$ and $z_n = 4$. The values in italics are those which may differ from $x_{l-1}^{(p)}$.

$l\backslash p$	0	1	2	3	4
0	000010010110111001	000010	1100	100	**100**
1	001011100110011010	000001	0100	000	000
2	010110101010011011	000010	1100	100	*001*
3	011011101110101101	000000	0001	*111*	*100*
4	101011101011110100	010001	*0000*	*000*	*000*
5	101011101011110101	*100000*	*1000*	*100*	*100*

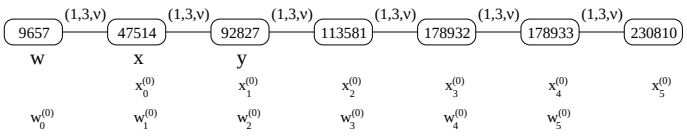

FIG. 2: A portion of the graph for the example given in Tables I and II. The vertices w, x, y, etc each have $i = 1$ and $j = 3$ for the edge labels, so it is necessary for the ν to differ to ensure that adjoining edges have distinct labels. The $x_l^{(0)}$ and $w_l^{(0)}$ which the vertices correspond to are also given. The numbers in the first columns of Tables I and II are the binary representations of the vertex numbers given here.

In Table I the values of $x_l^{(p)}$ are given, and the elements in the first column are values of $x_l^{(0)}$. As an example of calculation of $x_l^{(1)}$, note that $x_0^{(0)}$ differs from $x_1^{(0)}$ in the second bit position. The second bit for $x_0^{(0)}$ is 0, so this is the first bit for $x_0^{(1)}$. We subtract 1 from the bit position to obtain 1, and take the remaining bits of $x_0^{(1)}$ to be the binary representation of 1. For the case of $x_5^{(0)}$, this is the end of the chain, so we simply take $x_5^{(1)}$ to be the first bit of $x_5^{(0)}$, which is 1, and the binary representation of 0.

In Table II the values of $w_l^{(p)}$ are given, where these are calculated from a $w < x$ such that $x = f_y(w, i)$ and $w = f_y(x, j)$. The example given illustrates the case where the sequence of $w_l^{(0)}$ (with $w_l^{(0)} = x_{l-1}^{(0)}$) ends before the sequence of $x_l^{(0)}$. In this case, we find that the differences propagate towards the left, but we still have $x_0^{(z_n)} = w_1^{(z_n)}$. Thus different values of ν are obtained, as expected. For x we obtain $\nu = x_0^{(4)} = 000$, and for w we obtain $\nu = w_0^{(4)} = 100$.

We can use Lemma 2 to prove Theorem 2.

Proof. (of Theorem 2) Overall the number of Hamiltonians $H_{(i,j,\nu)}$ in the decomposition is $m = 6d^2$. To calculate $g(x, i, j, \nu)$, it is necessary to call the black-box $2(z_n + 2)$ times.

To simulate evolution under the Hamiltonian $H_{(i,j,\nu)}$, we require g to be implemented by a unitary operator U_g satisfying

$$U_g|x, i, j, \nu\rangle|0\rangle = |x, i, j, \nu\rangle|y, (H_{i,j,\nu})_{x,y}\rangle.$$

As discussed above, the function f may be represented by a unitary U_f; using this unitary it is straightforward to obtain a unitary $\tilde{U}_g$ such that

$$\tilde{U}_g|x, i, j, \nu\rangle|0\rangle = |\phi_{x,i,j,\nu}\rangle|y, (H_{i,j,\nu})_{x,y}\rangle.$$

We may obtain the unitary U_g in the usual way by applying $\tilde{U}_g$, copying the output, then applying $\tilde{U}_g^\dagger$ [20].

Using the method of Ref. [5], the Hamiltonian $H_{(i,j,\nu)}$ may be simulated using a call to U_g and a call to $U_g^\dagger$. As z_n is of order $\log^* n$, the number of black-box calls to f for the simulation of each $H_{(i,j,\nu)}$ is $O(\log^* n)$. Using these values, along with Eq. (1), we obtain the number of black-box queries as in Eq. (3). □

Another issue is the number of auxiliary operations, which is the number of operations that are required due to the overhead in calculating $\Upsilon(x, i, j)$. It is necessary to perform bit comparisons between a maximum of $z_n + 2$ numbers in the first step, and each has n bits. This requires $O(n \log^* n)$ operations. In the next step the number of bits is $O(\log_2 n)$ bits, which does not change the scaling. Hence the number of auxiliary operations is

$$O\left(n(\log^* n)^2 d^2 5^{2k} (d^2\tau)^{1+1/2k}/\epsilon^{1/2k}\right).$$

This scaling is superior to the scaling n^{10} in Ref. [5].

Next we consider the error introduced by calculating the matrix elements to finite precision. Given that the matrix elements are represented by $2n'$ bit integers, the error cannot exceed $\|H\|/2^{n'}$. The error in calculating $\exp(-iH_{(i,j,\nu)}t)$ will not exceed $\tau/2^{n'}$ [5], so the error in the integrator due to the finite precision does not exceed $4m5^k\tau/2^{n'}$. This error can then be kept below $\epsilon/2$ by choosing

$$n' > 5 + \log_2(\tau d^2 5^k/\epsilon).$$

The total error may be kept below ϵ by choosing the integrator such that the integration error does not exceed $\epsilon/2$.

VI. CONCLUSIONS

We have presented a scheme for simulating sparse Hamiltonians that improves upon earlier methods in two main ways. First, we have examined the use of higher order integrators to reduce the scaling to be close to linear in $\|H\|t$. Second, we have significantly improved the algorithm for the decomposition of the Hamiltonian, so the scaling of the number of black-box calls is close to $\log^* n$, rather than polynomial in n. In addition we have shown that the scaling cannot be sublinear in $\|H\|t$ (for reasonable values of n).

Acknowledgments

This project has been supported by the Australian Research Council, the University of Queensland, and Canada's NSERC, iCORE, CIAR and MITACS. R.C. thanks Andrew Childs for helpful discussions.

[1] Shor, P. W.: Algorithms for quantum computation: Discrete logarithms and factoring. Proc. 35th Symp. on Foundations of Computer Science, 124-134 (1994)
[2] Grover, L.: Quantum mechanics helps in searching for a needle in a haystack. Phys. Rev. Lett. **79**, 325-328 (1997)
[3] Feynman, R.P.: Simulating physics with computers. Int. J. Theoret. Phys. **21**, 467-488 (1982)
[4] Lloyd, S.: Universal quantum simulators. Science **273**, 1073-1078 (1996)
[5] Aharonov, D. and Ta-Shma, A.: Adiabatic quantum state generation and statistical zero knowledge. Proc. 35th Annual ACM Symp. on Theory of Computing, 20-29 (2003)
[6] Childs, A., Farhi, E., and Gutmann, S.: An example of the difference between quantum and classical random walks. J. Quant. Inf. Proc. **1**, 35-43 (2002)
[7] Shenvi, N., Kempe, J., and Whaley, K.B.: Quantum random-walk search algorithm. Phys. Rev. A **67**, 052307 (2003)
[8] Childs, A. and Goldstone, J.: Spatial search by quantum walk. http://arxiv.org/abs/quant-ph/0306054 (2003)
[9] Ambainis, A.: Quantum walk algorithm for element distinctness. http://arxiv.org/abs/quant-ph/0311001 (2003)
[10] Ambainis, A., Kempe, J., and Rivosh, A.: Coins make quantum walks faster. http://arxiv.org/abs/quant-ph/0402107 (2004)
[11] Farhi, E., Goldstone, J., Gutmann, S., and Sipser, M.: Quantum computation by adiabatic evolution. http://arxiv.org/abs/quant-ph/0001106 (2000)
[12] Suzuki, M.: Fractal decomposition of exponential operators with applications to many-body theories and Monte Carlo simulations. Phys. Lett. A **146**, 319-323 (1990)
[13] Suzuki, M.: General theory of fractal path integrals with applications to many-body theories and statistical physics. J. Math. Phys. **32**, 400-407 (1991)
[14] Childs, A.M.: Quantum information processing in continuous time. Ph.D. Thesis, Massachusetts Institute of Technology, 2004
[15] Cole, R. and Vishkin, U.: Deterministic coin tossing with applications to optimal parallel list ranking. Inform. and Control **70**, 32-53 (1986)
[16] Linial, N.: Locality in distributed graph algorithms. SIAM J. Comput. **21**, 193-201 (1992)
[17] Childs, A.M., Cleve, R., Deotto, E., Farhi, E., Guttman, S., and Spielman, D.A.: Exponential algorithmic speedup by quantum walk. Proc. 35th Annual ACM Symp. on Theory of Computing, 59-68 (2003)
[18] Ahokas, G.: Improved algorithms for approximate quantum Fourier transforms and sparse Hamiltonian simulations. M.Sc. Thesis, University of Calgary, 2004
[19] Beals, R., Buhrman, H., Cleve, R., Mosca, M., and de Wolf, R.: Quantum lower bounds by polynomials. J. ACM **48**, 778-797 (2001)
[20] Nielsen, M.A. and Chuang, I.L.: *Quantum Computation and Quantum Information*. Cambridge: Cambridge University Press, 2000
[21] For $z_n > 6$ we require $n > 10^{10^{37}}$; clearly an unrealistic problem size.