Parallel Quantum Algorithm for Hamiltonian Simulation

Zhicheng Zhang[1,2], Qisheng Wang[3,4], and Mingsheng Ying[5,4]

[1]Centre for Quantum Software and Information, University of Technology Sydney, Sydney, Australia
[2]University of Chinese Academy of Sciences, Beijing, China
[3]Graduate School of Mathematics, Nagoya University, Nagoya, Japan
[4]Department of Computer Science and Technology, Tsinghua University, Beijing, China
[5]State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China

We study how parallelism can speed up quantum simulation. A parallel quantum algorithm is proposed for simulating the dynamics of a large class of Hamiltonians with good sparse structures, called uniform-structured Hamiltonians, including various Hamiltonians of practical interest like local Hamiltonians and Pauli sums. Given the oracle access to the target sparse Hamiltonian, in both query and gate complexity, the running time of our parallel quantum simulation algorithm measured by the quantum circuit depth has a doubly (poly-)logarithmic dependence $\mathrm{polylog}\log(1/\epsilon)$ on the simulation precision ϵ. This presents an *exponential improvement* over the dependence $\mathrm{polylog}(1/\epsilon)$ of previous optimal sparse Hamiltonian simulation algorithm without parallelism. To obtain this result, we introduce a novel notion of parallel quantum walk, based on Childs' quantum walk. The target evolution unitary is approximated by a truncated Taylor series, which is obtained by combining these quantum walks in a parallel way. A lower bound $\Omega(\log\log(1/\epsilon))$ is established, showing that the ϵ-dependence of the gate depth achieved in this work cannot be significantly improved.

Our algorithm is applied to simulating three physical models: the Heisenberg model, the Sachdev-Ye-Kitaev model and a quantum chemistry model in second quantization. By explicitly calculating the gate complexity for implementing the oracles, we show that on all these models, the total gate depth of our algorithm has a $\mathrm{polylog}\log(1/\epsilon)$ dependence in the parallel setting.

Zhicheng Zhang: iszczhang@gmail.com, part of the work was done when the author was at the University of Chinese Academy of Sciences, Beijing, China.

Qisheng Wang: qishengwang1994@gmail.com, part of the work was done when the author was at the Department of Computer Science and Technology, Tsinghua University, Beijing, China.

Mingsheng Ying: yingms@ios.ac.cn yingmsh@tsinghua.edu.cn

Contents

1 Introduction

Simulating the quantum Hamiltonian dynamics is a fundamental problem in computational physics. Despite its ubiquity and importance, the problem is believed to be intractable for classical computers. Quantum computers were originally proposed to efficiently solve this problem [1]. The first algorithm for solving this problem was given by Lloyd for local Hamiltonians [2], and has been followed by many remarkable results over the past twenty years. Moreover, these results have found diverse applications in other quantum algorithms (e.g., [3–6]) beyond quantum simulation.

While the state-of-the-art has achieved an optimal quantum algorithmic solution to simulating a large class of Hamiltonians [7], it remains open whether the quantum simulation algorithms can be parallelized in order to provide further speed-up. In this work, we identify a class of Hamiltonians that can be more efficiently simulated in parallel, called uniform-structured Hamiltonians. Then we introduce the notion of parallel quantum walk within Childs' framework [3, 8–10]. Based on it, we propose a parallel quantum simulation algorithm for uniform-structured Hamiltonians.

Hamiltonian simulation

Simulating the time evolution of a quantum system governed by a time-independent Hamiltonian H for a time t is essentially approximating the unitary e^{-iHt} to some precision ϵ, according to the Schrödinger equation. In this paper, we focus on the digital quantum simulation (rather than the analog quantum simulation [11]), that is, simulating the Hamiltonian with a fault-tolerant universal quantum computer, given some oracle access to the Hamiltonian H. The complexity of a quantum simulation algorithm consists of query complexity and gate complexity, which depends on several factors: the simulation time t, the precision ϵ, and other parameters of the target Hamiltonian (e.g., size, matrix norm, and sparsity of H).

In the literature, there are basically three approaches to simulating a Hamiltonian:

- The *product formula approach* is conceptually the simplest without introducing ancilla qubits. Early works [2, 12–16] on product-formula-based algorithms often had a poor complexity dependence on the precision ϵ, which was later improved [17] by techniques borrowed from other simulation approaches. This approach has regained attention in recent years due to a number of new results (e.g., [17–29]) and its potential to be implemented in the near term.

- The *LCU (linear combination of unitaries)-based approach* [16] spawned the groundbreaking work [30] that improves the complexity dependence on ϵ from $\mathrm{poly}(1/\epsilon)$ to $\mathrm{polylog}(1/\epsilon)$ (which is also achieved earlier in [31] by fractional queries). The idea is to approximate e^{-iHt} with a linear combination of unitaries easier to implement. This approach can be used to simulating a d-sparse Hamiltonian H, which is an Hermitian matrix with at most d nonzero entries in each row. The Hamiltonian H is accessed by two oracles: an oracle $\mathcal{O}_H$ giving the entry H_{jk} according to the index pair (j,k), and an oracle $\mathcal{O}_L$ giving the column index of the c^{th} nonzero entry in row j according to c and j.

  In particular, combined with Childs' quantum walks [8, 9, 32], this approach achieves a nearly optimal [10] query complexity for simulating sparse Hamiltonians. The LCU technique has also been applied to solving the quantum linear systems problem [3], which was originally solved in [33] by phase estimation.

- The lower bound on query complexity for sparse Hamiltonian simulation was finally reached by the *quantum signal processing approach* [7, 34, 35], which provides a new way to transform the eigenvalue of a unitary by manipulating a single ancilla qubit without performing phase estimation. The input model was also generalized beyond sparse matrices by subsequent works on qubitization and block-encoding [6, 7, 36].

Later works make further improvements on the complexity dependence on other parameters [35, 37–40], and the average-case complexity [41]. We particularly note that almost all[1] of the above approaches for Hamiltonian simulation are *sequential*.

Parallel quantum computation

The aim of this paper is to study parallel quantum simulation of Hamiltonians. The computational model we adopt is the quantum circuit model where the running time of a quantum algorithm is measured by the depth of its circuit implementation, with both gates and oracle queries being allowed to be performed in parallel.

A pioneering work in this direction is Cleve and Watrous' $O(\log n)$-depth parallel quantum circuit for n-qubit quantum Fourier transform [42], which can be used to parallelize Shor's factoring algorithm [43] with poly-time classical pre- and post-processing. Following works on parallel Shor's algorithm include factoring on a 2D quantum architecture [44] and discrete logarithm on elliptic curves [45]. The limits on parallelizing Grover's search algorithm [46] were first considered by Zalka [47]. This inspired a line of studies [48–52] on parallel quantum query algorithms and complexity bounds. The low-depth (parallel) quantum circuit classes are also studied (e.g., [53–63]), amongst which one surprising result is a quantum advantage established by constant-depth quantum circuits over their classical counterparts [64–67]. Recently, [68] proposed a quantum algorithm with a constant quantum depth for multivariate trace estimation.

The research on parallel quantum computation is not restricted to the circuit model. For example, in measurement-based quantum computing, it was observed that parallelism can provide more benefits than in the circuit model [69–71]. Another parallel model closer to the current quantum hardware is distributed quantum computing, which can efficiently simulate the quantum circuit model with low depth overhead [72]. Parallelism is also studied at more abstract levels like quantum programming [73, 74].

1.1 Main Results

Our main result is a parallel quantum simulation algorithm for uniform-structured Hamiltonians, which will be formally defined in Sections 3.2 and 3.3. These Hamiltonians include *local Hamiltonians*, *Pauli sums* and other Hamiltonians of interest.[2] Roughly speaking, a uniform-structured Hamiltonian H acting on n qubits has the form $H = \sum_{w\in[m]} H_w$,[3] where $m = \mathrm{poly}(n)$, and for each w, H_w is a sparse Hamiltonian whose structure is specified by a parameter s_w. Here, the structure of a sparse Hamiltonian is basically a compact

[1] The only exception we know is [37]. The depth complexity of their algorithm for simulating time-dependent lattice Hamiltonians is smaller than the size complexity by an $O(n)$ factor, where n is the number of qubits.

[2] A local Hamiltonian can be actually represented as a Pauli sum, by decomposing each local term with respect to the Pauli basis. Instead of writing a local Hamiltonian as a Pauli sum, in this paper we adopt a more natural way to describe the local Hamiltonians, which still fits well within our framework of uniform-structured Hamiltonians.

[3] In this paper, $[m]$ denotes the set $\{0, 1, \ldots, m-1\}$ for all $m \in \mathbb{N}$.

way to describe how the non-zero entries are arranged in its matrix representation. For example, for a local Hamiltonian H with H_w being local terms, we can choose each s_w to be an n-bit string with l bits of 1 that indicate the subsystem of l qubits on which H_w non-trivially acts. A good observation of the sparse structure, i.e., a good choice of s_w, is definitely important for our algorithm. We adopt the sparse matrix input model for these Hamiltonians, that is, a target Hamiltonian H is accessed by two oracles: an oracle $\mathcal{O}_H$ giving an entry H_{jk} by the mapping $|j,k,0\rangle \mapsto |j,k,H_{jk}\rangle$, and an oracle $\mathcal{O}_P$ giving the parameter s_w by the mapping $|w,0\rangle \mapsto |w,s_w\rangle$. Here, $\mathcal{O}_P$ might be different for various types of Hamiltonians. Formal details of the input model will be described in Section 2.

Throughout this paper we assume the target Hamiltonian H is normalized such that $\|H\|_{\max} = 1$, where $\|H\|_{\max} := \max_{jk}|H_{jk}|$. Then our main result can be stated as the following:

Theorem 1.1 (Informal version of Theorem 5.5)**.** *Any uniform-structured Hamiltonian $H = \sum_{w\in[m]} H_w$ acting on n qubits with each H_w being d-sparse, can be simulated for time t to precision ϵ by a quantum circuit of depth* $\mathrm{poly}(\log\log(1/\epsilon), \log n, m, d, t)$ *and size* $\mathrm{poly}(\log(1/\epsilon), n, m, d, t)$.

Here, the running time of our algorithm, i.e., the quantum circuit depth, has a doubly (poly-)logarithmic dependence on the precision ϵ. To the best of our knowledge, this is the first Hamiltonian simulation algorithm that achieves such dependence on ϵ.

Applying this theorem (actually Theorem 5.5) to simulating local Hamiltonians, we have:

Corollary 1.2 (Parallel simulation of local Hamiltonians)**.** *Let $H = \sum_{w\in[m]} H_w$ be an l-local Hamiltonian*[4] *acting on n qubits, where each H_w acts on a subsystem of l qubits whose positions are indicated by l bits of 1 in an n-bit string s_w. Suppose the oracle $\mathcal{O}_P$ has access to s_w such that $\mathcal{O}_P|w,0\rangle = |w,s_w\rangle$. Then H can be simulated for time t to precision ϵ by a quantum circuit of*

- *depth $O(\tau\log\gamma)$ and size $O(\tau\gamma)$ w.r.t. queries*[5] *to $\mathcal{O}_H$,*
- *depth $O(\tau\log\gamma)$ and size $O(m\tau\gamma)$ w.r.t. queries to $\mathcal{O}_P$, and*
- *depth $O\Big(\tau\Big(\log^2\gamma\cdot\log^2 n + \log^3\gamma\Big)\Big)$ and size $O(\tau\gamma^2\cdot(mn^4+\gamma^3))$ w.r.t. gates,*

where $\tau := m2^l\cdot t$, $\gamma := \log(\tau/\epsilon)$, and we allow arbitrary one- or two-qubit gates.

The best known algorithm for this task is by applying the optimal sparse Hamiltonian simulation of [7, 35] (note that H in Corollary 1.2 is $\big(m2^l\big)$-sparse), which requires $O\Big(\tau + \frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\Big)$ queries and a factor $O(n + \log(1/\epsilon)\cdot\mathrm{polylog}(1/\epsilon))$ of additional gates. By introducing parallelism, our algorithm exponentially improves the dependence on ϵ, in the depth w.r.t. both queries and gates.

It is worth noting the difference between the oracle $\mathcal{O}_P$ used in Corollary 1.2 for local Hamiltonians and the oracle $\mathcal{O}_L$ in previous works [8–10, 12, 35] for generic sparse Hamiltonians. Oracle $\mathcal{O}_L$ computes a function $L(j,c)$ denoting the column index of the c^{th} non-zero

[4] In this paper, for an l-local Hamiltonian, l is not necessarily constant (as is usually assumed) and can depend on n (e.g., $l = \mathrm{polylog}(n)$). Also, the local Hamiltonian $H = \sum_w H_w$ is not necessarily *geometrically local*, which further requires each H_w to act on adjacent l qubits.

[5] For simplicity, the depth/size (complexity) w.r.t. queries/gates refers to the depth/size of the circuit composed of the specified queries/gates, respectively. See Section 2.1 for detailed definitions.

entry in row j of H. In practice, oracle $\mathcal{O}_P$ is a more natural choice than $\mathcal{O}_L$, because if one wants to exploit the local structure of the Hamiltonian to be simulated, knowing the locality parameter s_w given by $\mathcal{O}_P$ is intuitively the minimal requirement. As evidence, for example, when we apply our algorithm to the Heisenberg model in Section 7.1, it turns out that implementing the oracle $\mathcal{O}_P$ is gate-efficient. Note that in the local Hamiltonian case, a query to the oracle $\mathcal{O}_L$ can be achieved by at most $m2^l$ queries to $\mathcal{O}_P$.

Lower bounds

It was shown in [31] that sparse Hamiltonian simulation requires $\Omega\Big(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\Big)$ queries. We are able to further prove a lower bound on the gate depth for the Hamiltonian simulation:

Theorem 1.3 (ϵ-dependence depth lower bound for Hamiltonian simulation)**.** *Any quantum algorithm for sparse Hamiltonian simulation to precision ϵ has depth complexity $\Omega(\log\log(1/\epsilon))$ w.r.t. gates. The same holds even for uniform-structured Hamiltonian simulation.*

This theorem implies that our parallel quantum algorithm given in Theorem 1.1 for simulating uniform-structured Hamiltonians cannot be significantly improved in the ϵ-dependence.

Applications

For illustration, our algorithm is applied in Section 7 to simulating three quantum dynamical models in physics and chemistry of practical interest:

- The Heisenberg model for studying the self-thermalization and many-body localization [75–77];
- The Sachdev-Ye-Kitaev (SYK) model for studying the simplest AdS/CFT duality [78–81]; and
- A second-quantized molecular model for studying the electronic structure of a molecule [82–84].

We explicitly calculate the gate cost for implementing the oracles mentioned above and the total gate complexity for the simulation. Table 1 shows a comparison of our algorithm with previous best known algorithms on the same tasks. From it, one can see that by introducing parallelism, our algorithm achieves an exponential speed-up on the ϵ-dependence for simulating all these models.

It is however worth pointing out that the parallel speed-up in the above applications has limitations in practice, because the n-dependence in the depth complexity (w.r.t. gates) of our algorithm is worse than previous best algorithms in these cases. For simulating local Hamiltonians, as stated in Corollary 1.2, the n-dependence comes from the linear (up to a poly-logarithmic factor) dependence on m (the number of local terms) and $\|H\|_{\max}$ (the norm of the Hamiltonian). For example, in the SYK model, m scales as $O(n^4)$ and $\|H\|_{\max}$ scales as $O(n^{2.5})$, so our algorithm has n-dependence $\tilde{O}(n^{6.5})$. In comparison, the previous best algorithm [85] for simulating the SYK model has a sublinear dependence on m, by using asymmetric qubitization to exploit the structure of the Hamiltonian, which results in a better n-dependence $\tilde{O}(n^{3.5})$. Due to the above issue, in practical scenarios where the required precision is not too high, for example, $\epsilon = 2^{-O(n)}$, our improvement on the ϵ-dependence becomes less significant because the n-dependence is the dominant part.

Model	Algorithms	Gate depth (n, t)	Gate depth (ϵ)
Heisenberg model	Childs et al. [77]	$\tilde{O}(n^3)$	$O\left(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\right)$
	Haah et al. [37]	$\tilde{O}(n)$	$\mathrm{polylog}(1/\epsilon)$
	Our algorithm	$\tilde{O}(n^3)$	$O\left(\log^3\log(1/\epsilon)\right)$
SYK model	Babbush et al. [85]	$\tilde{O}(n^{3.5}t)$	$\mathrm{polylog}(1/\epsilon)$
	Our algorithm	$\tilde{O}(n^{6.5}t)$	$O\left(\log^3\log(1/\epsilon)\right)$
Molecular model	Babbush et al. [84]	$\tilde{O}(n^8t)$	$O\left(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\right)$
	Later improvements (e.g. [86–90])	$\mathrm{poly}(n,t)$	$\mathrm{polylog}(1/\epsilon)$
	Our algorithm	$\tilde{O}(n^8t)$	$O\left(\log^3\log(1/\epsilon)\right)$

Table 1: A comparison of our algorithm (Theorem 5.5) with previous best algorithms in simulating three physical models. Here, parameter n is the size of the system to be simulated,[6] t is the simulation time, and ϵ is the precision of simulation. The complexity of an algorithm is measured by the depth complexity w.r.t. gates, where for readability the dependence on different parameters are split. The notation $\tilde{O}(\cdot)$ denotes an asymptotic upper bound suppressing poly-logarithmic factors. For Heisenberg model, we follow the convention of taking $t = n$.

1.2 High-level Overview of the Algorithm

Our algorithm is motivated by the LCU (linear combination of unitaries)-based approach to Hamiltonian simulation [10, 16, 30]. The basic idea of this approach is to approximate the target unitary e^{-iHt} with a linear combination of unitaries easier to implement. In particular, [10] uses a Chebyshev series approximation $e^{-iHt} \approx \sum_r \alpha_r \mathcal{T}_r(H)$, where for each r, $\alpha_r \in \mathbb{C}$ is some appropriate coefficient, and $\mathcal{T}_r(x)$ is the r-degree Chebyshev polynomial. Each $\mathcal{T}_r(H)$ can be obtained by r steps of Childs' quantum walk [8, 9]. Then a linear combination of these quantum walks is performed by the LCU technique [10, 16, 32]. Essentially, this approach is sequential due to the fact that r steps of quantum walk require r sequential queries, and to achieve a total precision ϵ of the simulation, r should be as large as $\Theta(\log(1/\epsilon))$, inducing a logarithmic precision-dependence.

In this work, we introduce a parallel quantum walk which is implementable with a quantum circuit of *constant* depth w.r.t. queries, for a large class of Hamiltonians with good sparse structures — uniform-structured Hamiltonians. The parallel quantum walk is *not* a direct parallelization of Childs' quantum walk; instead it implements a monomial of H. We express the unitary e^{-iHt} as a Taylor series $e^{-iHt} \approx \sum_r \beta_r H^r$ (like in the previous work [30]), where each r-degree monomial H^r can be obtained (with a proper scaling factor) by an r-parallel quantum walk. These parallel quantum walks are then linearly combined in parallel by a technique described in Section 4, which exploits parallelism in the LCU algorithm to combine R terms with depth complexity $\mathrm{polylog}(R)$. Since there are about $O(\log(1/\epsilon))$ terms in the LCU to achieve a total precision ϵ of the simulation, the depth complexity of our parallel algorithm w.r.t. queries is roughly $\mathrm{polylog}\log(1/\epsilon)$, achieving a doubly (poly-)logarithmic precision-dependence.

[6] More specifically, n has different meanings in different models: for the Heisenberg model, it is the exact number of qubits on the spin chain; for the SYK model, it is a half of the number of Majorana fermions; and for the second-quantized molecular model, it is the number of spin orbitals.

Parallel quantum walk

The main ingredient in our parallel simulation algorithm is the parallel quantum walk. For an intuition of the algorithm, let us consider a very special case of uniform-structured Hamiltonians for example, a tensor product of Pauli matrices; that is, $H = \bigotimes_{t\in[n]} \sigma_t$ for $\sigma_t \in \{\mathbb{1}, X, Y, Z\}$ being Pauli matrices. Although H is a simple 1-sparse Hamiltonian, it suffices for an illustration of the main idea in our algorithm. To begin with, one can think of H as a weighted adjacency matrix of a 1-sparse graph H, then a step of Childs' quantum walk without parallelism is a "superposition version" of a classical random walk: it performs $|j\rangle \mapsto H|j\rangle = H_{jk}|k\rangle$ for all vertices j in H, where k is the unique neighbor of vertex j, and $H_{jk} \in \{1, i, -1, -i\}$.

For this special case, multiple steps of Childs' quantum walk can be directly implemented in parallel, as shown in the following. Observe that the graph H consists of 2^{n-1} pairs of vertices, among which for each pair (j, k), multiple steps of Childs' quantum walk simply alternate between the state $|j\rangle$ and $|k\rangle$ with an accumulated phase. If we can predict (compute) the destination of r steps of this walk and the accumulated phase by quantum circuits with depth significantly less than r, then these r steps can be implemented efficiently in parallel.

Note that for H, there is a "uniform" way to determine the unique neighbor of a vertex. Consider an n-bit string s that characterizes the diagonality of Pauli matrices in the sequence $\langle\sigma_t\rangle_{t\in[n]}$: s has its t^{th} bit being 0 if $\sigma_t \in \{\mathbb{1}, Z\}$ is diagonal, and being 1 if $\sigma_t \in \{X, Y\}$ is off-diagonal. Then it is easy to see that j, k are neighbors if and only if $k = j \oplus s$, where $\oplus$ is the bit-wise XOR operator. Assume s is given by an oracle $\mathcal{O}_P$, a step of Childs' quantum walk $|j\rangle \mapsto H_{jk}|k\rangle$ can be implemented by first querying $\mathcal{O}_P$ to calculate $k = j \oplus s$, followed by querying $\mathcal{O}_H$ to add the phase H_{jk}.

Now r sequential steps of the walk essentially perform the mapping $|j\rangle \mapsto H^r|j\rangle = H_{jk}^{\lceil r/2\rceil} H_{kj}^{\lfloor r/2\rfloor} |l\rangle$, where $l = j \oplus s \oplus \ldots \oplus s = j \oplus s^{\oplus r} \in \{j, k\}$ is the destination of these steps. We can perform this mapping in parallel through two stages. In the first stage, we compute the destination l: first query r oracles $\mathcal{O}_P$ simultaneously to compute r copies of s; then classically compute $s^{\oplus r}$ in a binary tree of depth $O(\log r)$, using the associativity of XOR. In the second stage, the accumulated phase $H_{jk}^{\lceil r/2\rceil} H_{kj}^{\lfloor r/2\rfloor}$ is computed in a similar fashion, by querying $\mathcal{O}_H$ in parallel combined with parallel classical computation.

By a standard technique in reversible computation [91], the above parallel classical computation can be easily converted to parallel quantum computation, inducing depth complexity $O(1)$ w.r.t. queries and depth complexity $O(\log r)$ w.r.t. gates.

Although for the above example, one can easily find faster ways (by noting that $s^{\oplus r}$ is either s or 0 depending on $r \bmod 2$) to implement multiple steps of Childs' quantum walk in parallel, we adopt the above two-stage procedure because it captures the main idea of our algorithm. In general, denote H the Hamiltonian to be simulated and H its corresponding graph. We define an r-parallel quantum walk that can be implemented in two stages, where the two oracles $\mathcal{O}_H$ and $\mathcal{O}_P$ are queried separately. The first stage is called *pre-walk* which, roughly speaking, prepares a superposition over all paths of length r generated from r steps of unweighted random walk on the graph H. This stage intuitively predicts all possible paths in r steps of quantum walks, and can be done efficiently in parallel, with only a constant depth complexity w.r.t. queries to $\mathcal{O}_P$, provided that the Hamiltonian H is uniform-structured. The second stage is called *re-weight*, which adjusts the weights (i.e., the quantum amplitudes) of the state prepared by pre-walk, according to the entry values H_{jk} given by the oracle $\mathcal{O}_H$. This stage does not depend on the structure of H, and can be done efficiently in parallel with a constant depth complexity

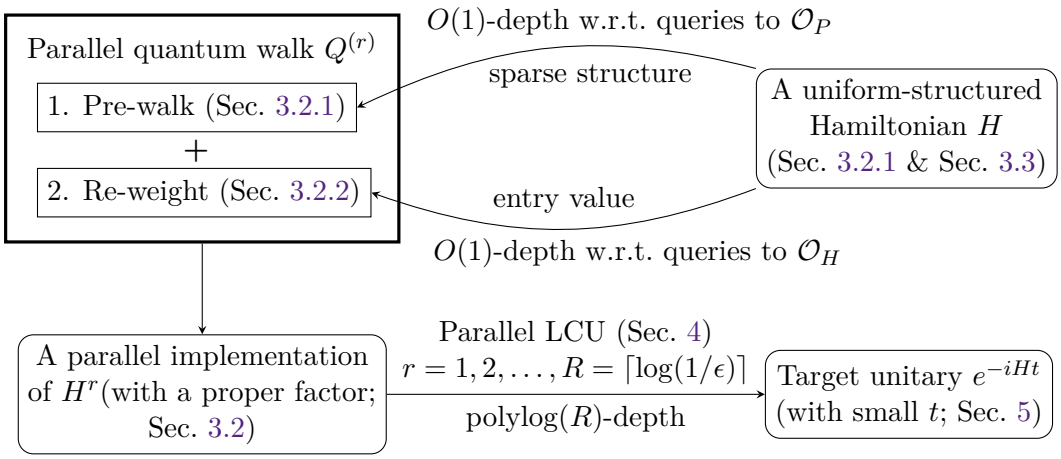

Figure 1: An outline of the parallel quantum algorithm for Hamiltonian simulation.

w.r.t. queries to $\mathcal{O}_H$. Finally combining with other techniques as in Childs' quantum walk, we can implement the monomial operator H^r (with a proper scaling factor, see Section 3 for details).

Note again that in the above example, although the parallel quantum walk performs multiple steps of Childs' quantum walk, in general they are *not* essentially equivalent. Specifically, r-parallel quantum walk implements H^r, while r steps of Childs' quantum walk yields $\mathcal{T}_r(H)$. For those readers familiar with block-encoding (see Definition 2.8), the parallel quantum walk actually provides a depth-efficient way to block-encode the operator H^r, if H is uniform-structured. It is more like an extension of a single step of Childs' quantum walk.

Parallel LCU for Hamiltonian series

To approximate the evolution unitary e^{-iHt} by a truncated Taylor series, the final step of our algorithm is to linearly combine the monomials H^r obtained from the parallel quantum walks discussed above. The ordinary LCU algorithm [3, 10] implementing a linear combination of R unitaries has depth complexity $\Theta(R)$. As pointed out in [3], if these unitaries are powers of a single unitary, then the LCU can be done in a parallel way analogous to the phase estimation [92] with depth complexity $O(\log R)$. We slightly generalize this result to implementing a linear combination of block-encoded (see Definition 2.8) powers of a Hamiltonian (called a Hamiltonian power series) in parallel. Since the LCU requires a state corresponding to the coefficients in the linear combination, we also present a parallel quantum algorithm for this state-preparation procedure, based on standard results in quantum sampling [93].

To summarize, the whole algorithm is visualized in Figure 1.

1.3 Related Works

Our algorithm for Hamiltonian simulation shows that by employing parallelism, the complexity dependence on the precision ϵ can be significantly reduced from polylog$(1/\epsilon)$ to polylog log$(1/\epsilon)$. For reducing the dependence on other parameters like the simulation time t, Atia and Aharonov [94] studied the fast-forwarding of Hamiltonians (which is further explored in a recent work [95]) — the ability to simulate a Hamiltonian by a quantum circuit with *size* significantly less than the simulation time t (e.g. polylog(t)). Note that the

concept of fast-forwarding does not concern parallelism and depth complexity. They show that the fast-forwarding of generic Hamiltonians is impossible unless BQP = PSPACE.[7] Nevertheless, they provided three examples of fast-forwardable Hamiltonians: Hamiltonians constructed from the modular exponentiation unitary in Shor's algorithm, commuting local Hamiltonians, and quadratic Hamiltonians. Interestingly, although these examples belong to the class of uniform-structured Hamiltonians defined in this paper (by properly setting the oracle $\mathcal{O}_H$ and $\mathcal{O}_P$), we are not aware of a direct way to extend our results to reduce the dependence on t by parallelization.

Jeffery, Magniez, and de Wolf [50] studied the parallel query complexity of element-distinctness and k-sum problems. The upper bounds in their results are obtained by what they called "parallelized quantum walk algorithms". But it should be noted that their algorithm is developed in the framework of finding marked elements via quantum walks [96], in particular, a modified quantum walk on multiple copies of Johnson graphs corresponding to a specific function (element-distinctness or k-sum), with parallel queries to that function. In contrast, our parallel quantum walk is defined in the framework of quantum walks for Hamiltonians [3, 8–10], in particular, a quantum walk that implements a monomial H^r (with a proper scaling factor) by r parallel queries to the oracles accessing a Hamiltonian H.

Recent Developments

After the work described in this paper, the depth complexity $O\Big(\tau\Big(\log^2\gamma\cdot\log^2 n+\log^3\gamma\Big)\Big)$ of our parallel quantum algorithm for local Hamiltonian simulation (see Corollary 1.2) was further improved to $O\Big(\tau\log^2\gamma\cdot\log^2 n\Big)$ in [97], which is achieved by depth-optimal quantum state preparation [97–100].

Recently, the depth lower bounds on Hamiltonian simulation were studied in [101]. They showed that simulating time-independent sparse Hamiltonians in general requires $\Omega(t)$ depth w.r.t. to queries; and based on the random oracle heuristic [102, 103], simulating time-independent local Hamiltonians and time-dependent geometrically local Hamiltonians requires $\Omega(t/n^c)$ depth w.r.t. gates, where c is some constant. This gives a negative answer to the second open question in Section 1.4 in the general case.

1.4 Discussion

In this paper, we propose a parallel quantum algorithm for Hamiltonian simulation that achieves a doubly (poly-)logarithmic precision-dependence in the depth complexity. This exponential improvement from the previous (poly-)logarithmic precision-dependence is rather a theoretical result, because in practice if the precision required is not too high, the improvement becomes less significant. Still, regarding the great importance of parallel computing and its success in the classical world, we hope our work may shed light on the design of parallel quantum algorithms and our ideas may contribute to future Hamiltonian simulation in practice.

Some readers might be also concerned that in order to extract classical information to the required precision from the output quantum state, the overhead of measurements could dominate the overall complexity. Indeed, our analysis only focuses on the asymptotic cost of the quantum simulation itself, without taking the cost of post-processing into

[7] Note that this differs from the well-known "no-fast-forwarding theorem" in [13] stating that no sparse Hamiltonian can be simulated for time t with sub-linear query complexity in t. Atia and Aharonov's result [94] is not restricted to the query model.

account. Nevertheless, one can expect that the output quantum state of our algorithm can be coherently fed to another quantum system (see [104–106]). For example, the output state of a quantum algorithm can be directly used as the input data in quantum machine learning [107], which is much more efficient than preparing the input data classically.

One direction of future work is to consider whether the complexity of Hamiltonian simulation can be further improved.

- Given simulation time t and precision ϵ, the depth complexity w.r.t. queries of our algorithm is $O(\tau \log \gamma)$, while the query complexity of the best sequential quantum algorithm [7, 35] is roughly $O(\tau + \gamma)$, where $\tau \propto t$ and $\gamma = \log(\tau/\epsilon)$. It remains open whether the multiplicative dependence on τ and γ can be improved to additive $O(\tau + \log \gamma)$.

- Another open question, as mentioned in Section 1.3, is whether parallelization can significantly improve the dependence on the simulation time t; more precisely, what kind of Hamiltonians can be simulated for time t by a quantum circuit of depth $\mathrm{polylog}(t)$, by allowing parallel queries?

- Intuitively, the power of parallelism is provided by the use of ancilla qubits. Considering the dependence on precision ϵ, from the upper bound on the circuit size in Theorem 1.1, we can derive a simple upper bound $O(\mathrm{polylog}(1/\epsilon))$ on the number of ancilla qubits in our algorithm. On the other hand, to achieve a significant speed-up (in particular, a circuit depth $O(\mathrm{polylog}\log(1/\epsilon))$), we can also derive a lower bound $\Omega\Big(\frac{\log(1/\epsilon)}{\mathrm{polylog}\log(1/\epsilon)}\Big)$ on the number of ancillae in simulating uniform-structured Hamiltonians. This is obtained by the fact that the size of a quantum circuit is no greater than the product of the circuit depth and the number of qubits, combined with the lower bound $\Omega\Big(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\Big)$ on the circuit size (see the proof of Theorem 1.3 and [7, 35]). Therefore, whether the number of ancillae can be further reduced remains an interesting question. More generally, the trade-off between the circuit depth and the number of ancillae will be an important issue in parallel quantum algorithms.

- As mentioned in Section 1.1, in several practical applications the depth complexity (w.r.t. gates) of our algorithm has worse dependence on the system size compared to the previous best algorithms. Is it possible to reduce the n-dependence in the complexity while retaining the improvement on the ϵ-dependence?

Another important direction is to study what other kinds of Hamiltonians with special structures can be simulated efficiently in parallel. Although the class of uniform-structured Hamiltonians in this work includes many physical Hamiltonians, one might also encounter non-physical ones in other quantum algorithms that use Hamiltonian simulation as a subroutine (e.g., quantum linear system solver [3, 33], and quantum differential equations solver [108]). How about exploiting the structures of those non-physical Hamiltonians for parallelism?

Finally, a very interesting attempt is to generalize our ideas and techniques to broader applications.

- In this paper we are only concerned about time-independent Hamiltonian simulation. For time-dependent Hamiltonian simulation, one approach [30, 38, 109–111] is to use the Dyson series to replace the Taylor series to approximate the unitary evolution. An interesting question is whether our algorithm (in particular, the parallel quantum walk) can be generalized for simulating time-dependent Hamiltonians.

- The major technical tool introduced in this work is a parallel quantum walk for Hamiltonians, which we believe can also be extended to general matrices. This tool might be applicable to other problems, for example, solving quantum linear differential equations [108], for which an operator e^{At} is to be implemented instead of the unitary e^{iHt} for Hamiltonian simulation, where A is a general matrix.

- As quantum singular value transform (QSVT) [36] has shown its power in manipulating the block-encodings of general matrices, and serves as a building block for many other quantum algorithms (e.g., [4, 112, 113]), it is also interesting to ask whether parallelism can provide speed-ups for QSVT.

1.5 Structure of the Paper

For convenience of the reader, some preliminaries are presented in Section 2. In Section 3, we introduce a parallel quantum walk for Hamiltonians. More concretely, we revisit the framework of Childs' quantum walk in Section 3.1, give a parallelization of it and show how to implement this parallel quantum walk in Section 3.2, and analyze the complexity in Sections 3.2.1 and 3.2.2. Specifically in Section 3.2.1, we define uniform-structured Hamiltonians, for which the parallel quantum walk can be performed efficiently. In Section 3.3 we present an extension of the parallel quantum walk to the case of a sum of sparse Hamiltonians, where we also extend the class of uniform-structured Hamiltonians to include more Hamiltonians of interest. In Section 4, we show how to implement a parallel LCU for a Hamiltonian power series. Section 5 assembles the above results to simulate a Hamiltonian by combining parallel quantum walks. In Section 6, we prove an ϵ-dependence lower bound on the gate depth for Hamiltonian simulation. In Section 7, we apply our algorithm to simulate three concrete physical models.

2 Preliminaries and Notations

2.1 Basic Terminologies

Sparse matrix input model

Let H be a d-sparse $N \times N$ Hamiltonian acting on n qubits with $\|H\|_{\max} = 1$. H is accessed by:

- An entry oracle $\mathcal{O}_H^b$ that gives an entry of H with b-bit precision such that

$$\mathcal{O}_H^b |j,k,z\rangle = |j,k,z \oplus H_{jk}\rangle ,$$

  for $j,k \in [N]$ and $z, H_{jk} \in [2^b]$, where the complex number H_{jk} is stored in a b-bit string that contains its real part and imaginary part (each with $b/2$ bits assuming b is even), and b will be determined by the precision ϵ of the algorithm. When we are just referring to the oracle we may omit the superscript b.

- A sparse structure oracle $\mathcal{O}_P$ that gives parameters about the sparse structure of H such that

$$\mathcal{O}_P |x,y\rangle = |x, P(x,y)\rangle$$

  for $x \in \mathcal{X}, y \in \mathcal{Y}$, where $\mathcal{X}, \mathcal{Y}$ are sets of integers, and $P : \mathcal{X} \times \mathcal{Y} \to \mathcal{Y}$ is a function determined by the sparse structure of H such that for all $x \in \mathcal{X}$, $P(x,\cdot)$ is a bijection for $y \in \mathcal{Y}$.

In many previous works [3, 8–10, 12] on sparse Hamiltonians, instead of $\mathcal{O}_P$, an oracle $\mathcal{O}_L$ is given such that $\mathcal{O}_L |j,t\rangle = |j, L(j,t)\rangle$, where $L(j,t) \in [N]$ gives the column index of the t^{th} nonzero entry in row j of H for $t \in [d]$ and gives 0 for $t \notin [d]$. We note that $\mathcal{O}_L$ can be expressed as a special case of $\mathcal{O}_P$ by taking $\mathcal{X} = \mathcal{Y} = [N]$ and $P = L$. However, compared to these work, we adopt a more general oracle $\mathcal{O}_P$ because Hamiltonians investigated in this paper have different sparse structures, which can be better exploited by different concrete forms of $\mathcal{O}_P$ (see examples in Sections 3.2 and 3.3). Moreover, the implementations of these $\mathcal{O}_P$'s turn out to be very efficient, compared to much costlier implementations of the oracle $\mathcal{O}_H$, as shown in Section 7 when we calculate the total gate complexity of our algorithm in simulating practical physical models.

As in previous works, we allow using (single-qubit) controlled versions of these oracles, and will not explicitly distinguish between the controlled and uncontrolled versions.

Complexity model

In this paper, we will consider both query complexity and gate complexity, where we allow arbitrary one- or two-qubit gates. For a parallel quantum algorithm represented by a quantum circuit, we will measure its cost by its depth and size. When considering the gate complexity, each oracle query is temporarily counted as one gate. The depth of a quantum circuit w.r.t. gates is defined as the length of the longest path composed of gates and wires from the circuit inputs to outputs, where the length is the total number of gates on this path. The size of a quantum circuit w.r.t. gates is defined as the total number of gates in the entire quantum circuit.

We adopt the definition of parallel query complexity in previous works (see for example [50]). The query complexity is calculated for each oracle separately. When considering the query complexity with respect to an oracle $\mathcal{O}$, all gates and queries to other oracles are ignored. We allow queries to multiple copies of $\mathcal{O}$ in parallel — that is, we can perform the mapping

$$|\psi_1\rangle \otimes \ldots \otimes |\psi_r\rangle \mapsto \mathcal{O} |\psi_1\rangle \otimes \ldots \otimes \mathcal{O} |\psi_r\rangle$$

in a single time-step for some r. From the viewpoint of a quantum circuit, these parallel queries to $\mathcal{O}$ act as "gates" in the same circuit layer. The depth of a quantum circuit w.r.t. queries to $\mathcal{O}$ is defined as the length of the longest path composed of queries to $\mathcal{O}$ and wires from the circuit inputs to outputs, where the length is the total number of queries to $\mathcal{O}$ on this path. The size of a quantum circuit w.r.t. queries to $\mathcal{O}$ is then defined as the number of total queries in the entire quantum circuit.

We use the following terminologies. The depth/size complexity of an algorithm (resp. problem) refers to the depth/size (w.r.t. queries or gates if specified) of quantum circuits implementing (resp. solving) it. Without specification, the depth/size (complexity) implicitly means the depth/size (complexity) w.r.t. gates. The query/gate depth refers to the depth complexity w.r.t. queries/gates. We will also use the adjectives "α-depth and β-size" to describe a circuit of depth α and size β (w.r.t. queries or gates if specified).

Error model

A pure state $|\psi\rangle$ is said to be approximated by $|\tilde{\psi}\rangle$ to precision ϵ, if they are close in the l_2-norm such that $\left\| |\psi\rangle - |\tilde{\psi}\rangle \right\| \leq \epsilon$. A unitary U is said to be implemented to precision ϵ, if a unitary $\tilde{U}$ is actually implemented such that $\left\| U - \tilde{U} \right\| \leq \epsilon$, where the norm is the spectral norm. The terms "ϵ-precise" and "ϵ-close" will also be used interchangeably.

2.2 Parallel Quantum Circuit

Now we review some basic techniques for constructing parallel quantum circuits. Although the results are known in the previous literature, we provide their proofs in order to illustrate basic ideas in designing parallel quantum circuits.

Lemma 2.1 (Parallel copying [42, 54])**.** *Let* COPY_b *be a unitary that creates* b *copies of a bit (including the original copy); that is,*

$$\mathrm{COPY}_b |x\rangle |0\rangle^{\otimes b-1} = |x\rangle^{\otimes b}$$

for all $x \in \{0,1\}$*. There is an* $O(\log b)$*-depth and* $O(b)$*-size quantum circuit that implements* COPY_b*.*

Proof. Suppose b is a power of two w.l.o.g. Then as shown in Figure 2, the gate COPY_b can be inductively constructed from $\mathrm{COPY}_{b/2}$, with COPY_1 being the identity operator. It is easy to check the depth and size of this quantum circuit by induction. □

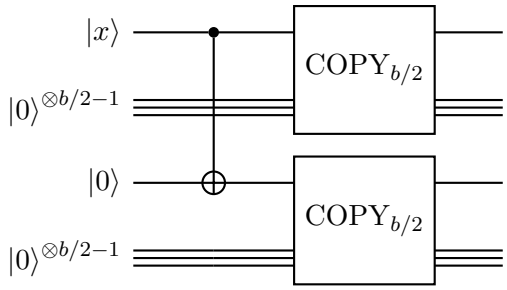

Figure 2: Inductive construction of the gate COPY_b

Lemma 2.1 can be easily extended from copying a single bit to copying an m-bit string, with the circuit depth and size multiplied by m. We use COPY_b^m to denote such a circuit. For the case $b = 2$, we will omit the subscript and write COPY^m.

Lemma 2.2 (Parallel controlled rotations [42, 54])**.** *Let* $\mathrm{C}_b\text{-}R_Z$ *be a unitary that performs a phase shift on a single qubit controlled by* b *qubits; that is,*

$$\mathrm{C}_b\text{-}R_Z |\gamma\rangle |\psi\rangle = |\gamma\rangle R_Z\left(2\pi\gamma \cdot 2^{-b}\right) |\psi\rangle \tag{1}$$

for all $\gamma \in [2^b]$ *and* $|\psi\rangle \in \mathbb{C}^2$*, where* $R_Z(\theta) = e^{-i\theta Z/2}$*. Then* $\mathrm{C}_b\text{-}R_Z$ *can be implemented by an* $O(\log b)$*-depth and* $O(b)$*-size quantum circuit.*

The same conclusion holds for $\mathrm{C}_b\text{-}R_X$ *and* $\mathrm{C}_b\text{-}R_Y$ *defined by replacing* $R_Z(\theta)$ *in* (1) *with* $R_X(\theta) = e^{-i\theta X/2}$ *and* $R_Y(\theta) = e^{-i\theta Y/2}$*, respectively.*

Proof. Let γ_j represent the j^{th} bit of γ for $j \in [b]$. A parallel quantum circuit for $\mathrm{C}_b\text{-}R_Z$ is shown in Figure 3. To see its correctness, suppose $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, then $\mathrm{COPY}_b |\psi\rangle |0\rangle^{\otimes b-1} = \alpha|0\rangle^{\otimes b} + \beta|1\rangle^{\otimes b}$ is an entangled state. Since applying R_Z to any of the b entangled qubits will add to the relative phase, the whole state becomes $|\gamma\rangle \left(\alpha|0\rangle^{\otimes b} + e^{i2\pi\gamma\cdot 2^{-b}}\beta|1\rangle^{\otimes b}\right)$ after applying $\mathrm{C}\text{-}R_Z$ in parallel. The final state is obtained by reverse computation with $\mathrm{COPY}_b^\dagger$. It is easy to check the depth and size of the quantum circuit by Lemma 2.1.

The cases of X and Y-rotations can be easily proved by combining the above proof and the identities $R_X(\theta) = Had\, R_Z(\theta)\, Had$ and $R_Y(\theta) = R_Z(\pi/2)R_X(\theta)R_Z(\pi/2)$, where *Had* stands for the Hadamard gate. □

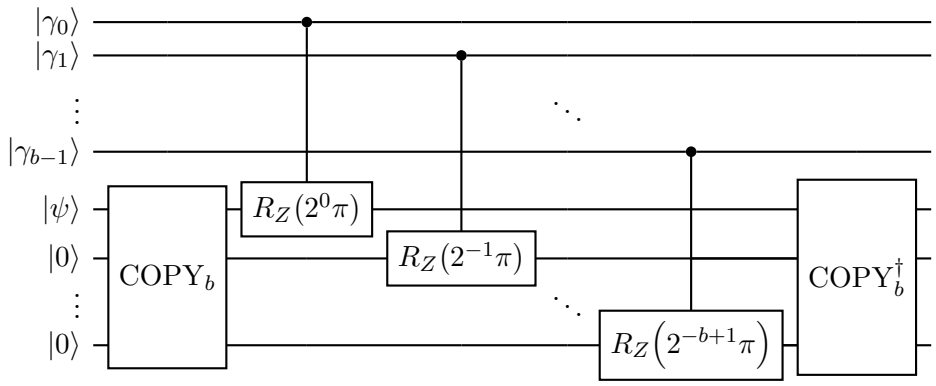

Figure 3: A parallel quantum circuit for C_b-R_Z

The following lemma accommodates a classical parallel computing technique into the quantum setting.

Lemma 2.3 (Sequence of associative operators)**.** *Let $\circ$ be an associative operator. Given a unitary $U_\circ$ that performs $|x, y, z\rangle \mapsto |x, y, z \oplus (x \circ y)\rangle$ and its inverse $U_\circ^\dagger$, then for all $m \in \mathbb{N}$, the mapping*

$$|x_1, \ldots, x_m\rangle |z\rangle \mapsto |x_1, \ldots, x_m\rangle |z \oplus (x_1 \circ \ldots \circ x_m)\rangle$$

can be implemented by a quantum circuit of depth $O(\log m)$ and size $O(m)$ w.r.t. $U_\circ$ and its inverse, where the additional gate complexity is often negligible compared to $U_\circ$ and thus omitted.

Proof. Let the partial sum $S_\circ(l, r) := x_l \circ \ldots \circ x_r$. Our goal is to compute $S_\circ(1, m) = x_1 \circ \ldots \circ x_m$. We can compute $S_\circ(l, r)$ inductively by

$$S_\circ(l, r) = S_\circ(l, p) \circ S_\circ(p + 1, r),$$

where $p = (l + r - 1)/2$. Assuming m is a power of two w.l.o.g., then the computation of $S_\circ(1, m)$ forms a tree of depth $O(\log m)$ and size $O(m)$ w.r.t. $U_\circ$, where the root is $S_\circ(1, m)$ and the leaves are $S_\circ(i, i) = x_i$ for $i \in [m]$. Once $S_\circ(1, m)$ is computed into an ancilla space, apply COPY gate to copy the result into $|z\rangle$, then clean all the garbage partial sums by reverse computation with $U_\circ^\dagger$. □

Corollary 2.4 (Parallel addition of a sequence [42])**.** *The addition of a sequence, i.e., the mapping*

$$|x_1, \ldots, x_m\rangle |z\rangle \mapsto |x_1, \ldots, x_m\rangle |z \oplus (x_1 + \ldots + x_m)\rangle$$

for all $x_1, \ldots, x_m, z \in [2^b]$, can be implemented by an $O(\log m + \log b)$-depth and $O(mb)$-size quantum circuit, where the addition is in $\mathbb{Z}_{2^b}$.

Proof. By Lemma 2.3, combined with classical techniques of three-two adder, pairwise representation and carry-lookahead adder. A detailed proof can be found in [42]. □

Corollary 2.5 (Parallel controlled Z gate)**.** *Let C_b-Z be a unitary that perform Z gate on a single qubit controlled by b qubits; that is,*

$$\mathrm{C}_b\text{-}Z |x\rangle |\psi\rangle = \begin{cases} |x\rangle Z |\psi\rangle, & x = 2^b - 1, \\ |x\rangle |\psi\rangle, & o.w. \end{cases}$$

for all $x \in [2^b]$ and $|\psi\rangle \in \mathbb{C}^2$, then C_b-Z can be implemented by an $O(\log b)$-depth and $O(b)$-size quantum circuit.

Proof. Write x as a bit string $x = x_0 \dots x_{b-1}$, then the C_b-Z gate can be implemented in the following ways. First take $\circ$ to be AND gate in Lemma 2.3 to compute $x_0 \wedge \dots \wedge x_{b-1}$ in the ancilla space by an $O(\log b)$-depth and $O(b)$-size quantum circuit, then conditioned on this apply a C-Z gate to $|\psi\rangle$, and finally clean the garbage by reverse computation. □

The following lemma translates the parallel classical results of computing elementary arithmetic functions [114] to the quantum case using the technique of reversible computing [91].

Lemma 2.6 (Parallel quantum circuit for elementary arithmetics [114])**.** *Let f be one of the following elementary arithmetic functions: addition, subtraction, multiplication, division, cosine, sine, arctangent, exponentiation, logarithm, maximum, minimum, factorial.*[8] *Then the unitary that performs*

$$|\tilde{x}\rangle |\tilde{y}\rangle |z\rangle \mapsto |\tilde{x}\rangle |\tilde{y}\rangle |z \oplus \tilde{f}(\tilde{x}, \tilde{y})\rangle$$

for all $\tilde{x}, \tilde{y}, z \in [2^b]$ can be implemented by an $O\left(\log^2 b\right)$-depth and $O(b^4)$-size quantum circuit,[9] *where $\tilde{x}, \tilde{y}$ are floating number representation of x, y on suitable intervals, and $\tilde{f}(\tilde{x}, \tilde{y})$ is 2^{-b}-close to $f(x, y)$. (For unary function the second operand y is omitted, e.g., $f(x, y) = cos(x)$.) In particular, for f being addition, subtraction or multiplication, the depth can be $O(\log b)$.*

In the remainder of this paper, the name "elementary arithmetic function" may also refer to a composition of a constant number of the arithmetic functions in Lemma 2.6. As elementary arithmetic operations are frequently used in this paper, in some cases we will measure the efficiency of computation with respect to these building blocks. More specifically we have the following definition for *arithmetic-depth-efficient* computation.[10]

Definition 2.7 (Arithmetic-depth-efficient computation)**.** A quantum circuit on b qubits is called arithmetic-depth-efficient if it is $O\left(\log^2 b\right)$-depth and $O(b^4)$-size. A function f is *arithmetic-depth-efficiently computable* if the mapping $|x\rangle |z\rangle \mapsto |x\rangle |z \oplus f(x)\rangle$ for all $x, z \in [2^b]$[11] can be implemented by an arithmetic-depth-efficient quantum circuit.

2.3 Block-encoding

Block-encoding is a recently introduced fundamental tool for arithmetic operations on matrices represented as a block of a unitary. It has been developed through a series of researches in quantum algorithms [3, 4, 6, 7, 10, 33, 36].

[8] In particular, for factorial, x is required to be $O\left(b^2\right)$.

[9] The complexity in [114] is more refined and complicated. Here we take a simple upper bound.

[10] Note that this differs from the notion of *depth-efficient* computation, which often refers to a poly-logarithmic depth computation. As an analog of the classical depth-efficient class NC, [54] introduced the quantum depth-efficient class QNC.

[11] Functions (or arithmetic operators) with multiple inputs and outputs can be easily converted to this form.

Definition 2.8 (Block-encoding)**.** An $(s+a)$-qubit unitary U is an (α, a, ϵ)-block-encoding of an s-qubit operator A if

$$\left\| A - \alpha\Big(\langle 0|^{\otimes a} \otimes \mathbb{1}\Big) U \Big(|0\rangle^{\otimes a} \otimes \mathbb{1}\Big) \right\| \leq \epsilon.$$

Intuitively, this implies the $2^s \times 2^s$ upper-left block of U is (ϵ/α)-close to A/α; that is,

$$U = \begin{pmatrix} B & * \\ * & * \end{pmatrix} \qquad \text{and} \qquad \|B - A/\alpha\| \leq \epsilon/\alpha.$$

In this paper, we will slightly abuse the terminology in such a way that if the condition is

$$\left\| A - \alpha\Big(\mathbb{1} \otimes \langle 0|^{\otimes a}\Big) U \Big(\mathbb{1} \otimes |0\rangle^{\otimes a}\Big) \right\| \leq \epsilon,$$

then for simplicity U is also called an (α, a, ϵ)-block-encoding of A.

3 Parallel Quantum Walk

In this section, we define a parallel quantum walk within the framework of Childs' quantum walks [3, 8–10]. In Section 3.1, we revisit Childs' quantum walk. Then we propose a parallelization of it in Section 3.2 and show how to implement the parallel quantum walk in two stages: pre-walk and re-weight, whose complexities are analyzed in Section 3.2.1 and Section 3.2.2 respectively. In particular, in Section 3.2.1 we define uniform-structured Hamiltonians for which the parallel quantum walk can be implemented efficiently. The Hamiltonians considered in Section 3.2 are d-sparse. In Section 3.3, we further consider a sum of d-sparse Hamiltonians and extend the parallel quantum walk and the class of uniform-structured Hamiltonians.

3.1 A Quantum Walk for Hamiltonians

Let H be a d-sparse $N \times N$ Hamiltonian acting on n qubits (i.e. $N = 2^n$). By analogy with the classical Markov chain, the Hermitian H can be seen as a transition matrix with "complex probability" on a d-sparse undirected graph whose adjacency matrix is given by replacing each nonzero entry in H with 1. Following [15], this graph is called *the graph of the Hamiltonian*, which we will often denote as H in the serif font throughout this paper. We write $(j,k) \in \mathsf{H}$ if an undirected edge (j,k) exists in the graph H. Following Childs' extension [3, 8–10, 32] of Szegedy's quantum walk [115], we define for all $j \in [N]$ the post-transition state of $|j\rangle$:

$$|\psi_j\rangle := \frac{1}{\sqrt{d}} \sum_{(j,k)\in\mathsf{H}} \left(\sqrt{H_{jk}^*}\,|k\rangle + \sqrt{1 - |H_{jk}|}\,|k+N\rangle \right), \tag{2}$$

as a generalization of the classical random walk, where z^* stands for the complex conjugate of z, and the square root $\sqrt{H_{jk}^*}$ is chosen such that $\sqrt{H_{jk}^*}(\sqrt{H_{jk}})^* = H_{jk}^*$. Note that $|\psi_j\rangle \in \mathbb{C}^{2N}$ because $N \leq k+N \leq 2N-1$ for $k \in [N]$. The factor $1/\sqrt{d}$ and the garbage states $|k+N\rangle$ are introduced to keep $|\psi_j\rangle$ a normalized state. Now we are ready to define the quantum walk, a unitary acting on the extended space $\mathbb{C}^{2N} \otimes \mathbb{C}^{2N}$ for the $N \times N$ Hamiltonian H.

Definition 3.1 (Quantum walk for Hamiltonians [3])**.** Given Hamiltonian H as above. Let $\mathcal{H} = \mathbb{C}^{2N} \otimes \mathbb{C}^{2N}$ be the state space. For each $j \in [N]$, define

$$|\Psi_j\rangle := |j\rangle \otimes |\psi_j\rangle = \frac{1}{\sqrt{d}} \sum_{(j,k)\in \mathsf{H}} |j\rangle \left(\sqrt{H^*_{jk}}\, |k\rangle + \sqrt{1 - |H_{jk}|}\, |k+N\rangle \right), \tag{3}$$

where $|\psi_j\rangle$ is defined in (2). Let $T : \mathcal{H} \to \mathcal{H}$ be any unitary such that

$$T(|j\rangle \otimes |z\rangle) = \begin{cases} |\Psi_j\rangle, & \text{if } j \in [N] \text{ and } z = 0, \\ \text{any state}, & o.w. \end{cases} \tag{4}$$

for all $j, z \in [2N]$. Let $S : \mathcal{H} \to \mathcal{H}$ be the SWAP operator such that $S\,|a, b\rangle = |b, a\rangle$ for all $a, b \in [2N]$. Then a step of quantum walk for H is defined as $Q := S(2\Pi_T - \mathbb{1})$, where $\Pi_T = T(|0\rangle\langle 0| \otimes \mathbb{1}_{n+1})T^\dagger$ is a projector, with $|0\rangle \in \mathbb{C}^{2N}$ and $\mathbb{1}_{n+1}$ the identity operator acting on $n + 1$ qubits.

The following lemma from [3] shows that we can implement a polynomial of H by multiple steps of quantum walk. More precisely, r iterations of Q block-encodes a r-degree Chebyshev polynomial of H.

Lemma 3.2. *$T^\dagger Q^r T$ is a $(1, n+2, 0)$-block-encoding of $\mathcal{T}_r(H/d)$, if T in (4) is performed precisely, where $\mathcal{T}_r(x)$ is the r-degree Chebyshev polynomial (of the first kind).*

The proof of Lemma 3.2 involves some interesting techniques, which were later used in [7] to develop qubitization. Here we only give a proof for the special case when $r = 1$ (thus $T^\dagger QT = T^\dagger ST$), which provides a basis for our generalization to parallel quantum walks in Section 3.2.

Corollary 3.3. *$T^\dagger ST$ is a $(1, n+2, 0)$-block-encoding of H/d, if T in (4) is performed precisely.*

Proof. To see

$$\left(\mathbb{1} \otimes \langle 0|^{\otimes n+2}\right) T^\dagger ST \left(\mathbb{1} \otimes |0\rangle^{\otimes n+2}\right) = H/d,$$

it suffices to prove

$$\left(\langle j| \otimes \langle 0|^{\otimes n+2}\right) T^\dagger ST \left(|l\rangle \otimes |0\rangle^{\otimes n+2}\right) = H_{jl}/d. \tag{5}$$

We first write the state $|\Psi_j\rangle$ in (3) into two parts $|\Psi_j\rangle = |\Phi_j\rangle + |\Phi_j^\perp\rangle$, where the subnormalized states

$$|\Phi_j\rangle := \frac{1}{\sqrt{d}} \sum_{(j,k)\in \mathsf{H}} |j\rangle \sqrt{H^*_{jk}}\, |k\rangle \quad \text{and} \quad |\Phi_j^\perp\rangle := \frac{1}{\sqrt{d}} \sum_{(j,k)\in \mathsf{H}} |j\rangle \sqrt{1 - |H_{jk}|}\, |k+N\rangle$$

with $|j\rangle, |k\rangle \in \mathbb{C}^{2N}$. It is easy to verify that

$$\begin{aligned} &\langle \Phi_j|\, S\, |\Phi_l\rangle = H_{jl}/d \\ &\langle \Phi_j|\, S\, |\Phi_l^\perp\rangle = \langle \Phi_j^\perp|\, S\, |\Phi_l\rangle = \langle \Phi_j^\perp|\, S\, |\Phi_l^\perp\rangle = 0 \end{aligned}$$

for all $j, l \in [N]$. Thus the LHS of (5) is $\langle \Psi_j|\, S\, |\Psi_l\rangle = H_{jl}/d$. □

Now we briefly illustrate how to implement one step of quantum walk by a quantum circuit. For the sake of simplicity we only consider query complexity, and assume each step is performed precisely. It suffices to show how to implement the unitary T (and thus $T^\dagger$), because in $Q = ST(2\left|0\right\rangle\left\langle 0\right| \otimes \mathbb{1}_{n+1} - \mathbb{1}_{2n+2})T^\dagger$ only the operator T requires oracle queries. As the ordinary quantum walk does not assume any special structure of the sparse Hamiltonian H (e.g., assume H is local), we take $\mathcal{O}_P = \mathcal{O}_L$ here.

Lemma 3.4 (Query complexity of T [10])**.** *The unitary T can be implemented by a quantum circuit with $O(1)$ queries to $\mathcal{O}_H$ and $\mathcal{O}_L$.*

Proof. Let $\mathcal{H}^A \otimes \mathcal{H}^B$ be the state space with $\mathcal{H}^A = \mathcal{H}^B = \mathbb{C}^{2N}$. As in the definition (4) of T, we only consider its action on the initial state $\left|j\right\rangle \otimes \left|0\right\rangle$ for $j \in [N]$, with $\left|j\right\rangle, \left|0\right\rangle \in \mathbb{C}^{2N}$. Then T can be implemented in the following way:

1. Prepare a uniform superposition over computational basis states of size d in $\mathcal{H}^B$.
2. Query the oracle $\mathcal{O}_L$ to obtain in $\mathcal{H}^B$ a superposition over nonzero entries in row j.
3. Query the oracle $\mathcal{O}_H$ to compute H_{jk} in an ancilla space, conditioned on which rotates the state in $\mathcal{H}^B$, then uncompute H_{jk} by reverse computation.

That is,

$$\begin{aligned}
\left|j\right\rangle \left|0\right\rangle &\xrightarrow{1} \left|j\right\rangle \frac{1}{\sqrt{d}} \sum_{t\in[d]} \left|t\right\rangle \\
&\xrightarrow{2} \left|j\right\rangle \frac{1}{\sqrt{d}} \sum_{(j,k)\in \mathsf{H}} \left|k\right\rangle \\
&\xrightarrow{3} \left|j\right\rangle \frac{1}{\sqrt{d}} \sum_{(j,k)\in \mathsf{H}} \left|k\right\rangle \left|H_{jk}\right\rangle \\
&\xrightarrow{3} \left|j\right\rangle \frac{1}{\sqrt{d}} \sum_{(j,k)\in \mathsf{H}} \left(\sqrt{H_{jk}^*}\left|k\right\rangle + \sqrt{1-\left|H_{jk}\right|}\left|k+N\right\rangle\right) \left|H_{jk}\right\rangle \\
&\xrightarrow{3} \left|j\right\rangle \left|\psi_j\right\rangle .
\end{aligned}$$

□

3.2 Parallelization

The algorithm for the quantum walk described in Section 3.1 is highly sequential, because r steps of quantum walk need r iterations of Q, which in total requires $\Theta(r)$ sequential queries to $\mathcal{O}_H$ and $\mathcal{O}_L$. Now we define a parallel quantum walk, which can be implemented by a quantum circuit of only constant depth w.r.t. queries for uniform-structured Hamiltonians (to be defined in Section 3.2.1). Slightly abusing the notation, we denote $\boldsymbol{j} \in \mathsf{H}^r$ if a path $\boldsymbol{j} = (j_0, \ldots, j_r)$[12] of length $r+1$ exists in the graph H.

Definition 3.5 (r-parallel quantum walk for Hamiltonians)**.** Given Hamiltonian H as above. Let $\mathcal{H} = \mathcal{H}^A \otimes \mathcal{H}^B$ be the state space, where $\mathcal{H}^A = \mathcal{H}^B = \left(\mathbb{C}^{2N}\right)^{\otimes(r+1)}$. For each

[12] In this paper, a vector is written in a bold font. The coordinates will be indexed from 0, for example, $\boldsymbol{a} = (a_0, \ldots, a_{k-1})$, where the dimension k is indicated by the context.

$j_0 \in [N]$, define

$$|\Psi_{j_0}^{(r)}\rangle := \frac{1}{\sqrt{d^r}} \sum_{\boldsymbol{j} \in \mathsf{H}^r} \underbrace{|\boldsymbol{j}\rangle}_{\in \mathcal{H}^A} \otimes \underbrace{|j_0\rangle \bigotimes_{s\in[r]} \left(\sqrt{H^*_{j_s j_{s+1}}} \, |j_{s+1}\rangle + \sqrt{1 - |H_{j_s j_{s+1}}|} \, |j_{s+1} + N\rangle \right)}_{\in \mathcal{H}^B} \tag{6}$$

where $|\boldsymbol{j}\rangle = |j_0\rangle \dots |j_r\rangle \in \mathcal{H}^A$. Let:

- $T^{(r)} : \mathcal{H} \to \mathcal{H}$ be any unitary operator such that

$$T^{(r)}(|j\rangle \, |z_1\rangle \dots |z_{2r+1}\rangle) = \begin{cases} |\Psi_j^{(r)}\rangle, & j \in [N], z_1 = \dots = z_{2r+1} = 0, \\ \text{any state}, & o.w. \end{cases} \tag{7}$$

  for all $j, z_1, \dots, z_{2r+1} \in [2N]$;

- $S^{(r)} : \mathcal{H} \to \mathcal{H}$ be the reverse order operator such that

$$S^{(r)} \, |a_0, \dots, a_{2r+1}\rangle = |a_{2r+1}, \dots, a_0\rangle$$

  for all $a_s \in [2N], s \in [2r+2]$.

Then a step of r-parallel quantum walk for H is defined as $Q^{(r)} := T^{(r)\dagger} S^{(r)} T^{(r)}$.

The parallel quantum walk defined above naturally generalizes the original quantum walk in Definition 3.1. The key idea is that we extend the state $|\Psi_j\rangle$ in (3) which is a superposition of one step of walk $(j,k) \in \mathsf{H}$, to the state $|\Psi_{j_0}^{(r)}\rangle$ in (6) which is a superposition of r steps of walk (i.e., a path) $\boldsymbol{j} \in \mathsf{H}^r$. As shown later in Lemma 3.8, the walk operator $Q^{(r)}$ becomes a block-encoding of the monomial $(H/d)^r$, a generalization of the H/d obtained from the original quantum walk. (In this sense, the walk operator $Q^{(r)}$ is an extension of $T^\dagger Q T$ instead of Q.) It should be emphasized that an r-parallel quantum walk is not equivalent to r sequential steps of the original quantum walk, which instead block-encodes a Chebyshev polynomial $\mathcal{T}_r(H/d)$.

Remark 3.6. The term "parallel quantum walk" comes from the fact that, as proved later, the walk operator $Q^{(r)}$ can be performed by a parallel quantum circuit with a constant query depth if the Hamiltonian H is uniform-structured (see Definition 3.10 in Section 3.2.1). This result is non-trivial, because the state $|\Psi_{j_0}^{(r)}\rangle$ in Definition 3.5 contains a dependence chain (which has a sequential nature) induced by the path $\boldsymbol{j} \in \mathsf{H}^r$, where j_{s+1} depends on j_s for all $l \in [r]$. This difficulty is resolved by observing that queries to oracle $\mathcal{O}_H$ can be actually separated from the dependence chain (see Section 3.2.2), while queries to oracle $\mathcal{O}_P$ can be parallelized if the graph H has a good structure (see Section 3.2.1).

Now we will illustrate why $Q^{(r)}$ is a block-encoding of $(H/d)^r$. Similar to the proof of Lemma 3.3, we can write $|\Psi_{j_0}^{(r)}\rangle$ into two parts $|\Psi_{j_0}^{(r)}\rangle = |\Phi_{j_0}^{(r)}\rangle + |\Phi_{j_0}^{(r)\perp}\rangle$, where the subnormalized state

$$|\Phi_{j_0}^{(r)}\rangle := \frac{1}{\sqrt{d^r}} \sum_{\boldsymbol{j} \in \mathsf{H}^r} |\boldsymbol{j}\rangle \otimes \sqrt{H^*_{j_0 j_1} \dots H^*_{j_{r-1} j_r}} \, |\boldsymbol{j}\rangle \tag{8}$$

represents the "good" part of $|\Psi_{j_0}^{(r)}\rangle$. The following lemma shows some orthogonal relations between these subnormalized states in the context of $S^{(r)}$.

Lemma 3.7. *For all $j, k \in [N]$, we have:*

1. $\langle \Phi_j^{(r)} | \, S^{(r)} \, | \Phi_k^{(r)} \rangle = ((H/d)^r)_{jk}$.
2. $\langle \Phi_j^{(r)} | \, S^{(r)} \, | \Phi_k^{(r)\perp} \rangle = \langle \Phi_j^{(r)\perp} | \, S^{(r)} \, | \Phi_k^{(r)\perp} \rangle = 0$.

Proof. We prove the two cases separately.

1. Let $j_0 := j$ and $k_0 := k$. By straightforward calculation we have:

$$S^{(r)} \, | \Phi_k^{(r)} \rangle = \frac{1}{\sqrt{d^r}} \sum_{\boldsymbol{k} \in \mathsf{H}^r} \sqrt{H^*_{k_0 k_1} \dots H^*_{k_{r-1} k_r}} \, | k_r \dots k_0 \rangle \, | k_r \dots k_0 \rangle$$

$$\langle \Phi_j^{(r)} | = \frac{1}{\sqrt{d^r}} \sum_{\boldsymbol{j} \in \mathsf{H}^r} \sqrt{H_{j_0 j_1} \dots H_{j_{r-1} j_r}} \, \langle j_0 \dots j_r | \, \langle j_0 \dots j_r | \, .$$

   Then using the self-adjointness of H we obtain:

$$\langle \Phi_j^{(r)} | \, S^{(r)} \, | \Phi_k^{(r)} \rangle = \frac{1}{d^r} \sum_{\boldsymbol{j} \in \mathsf{H}^r} H_{j_0 j_1} \dots H_{j_{r-1} j_r} = \left(\frac{H^r}{d^r} \right)_{jk} .$$

2. Recall that our state space is $\mathcal{H} = \mathcal{H}^A \otimes \mathcal{H}^B$ with $\mathcal{H}^A = \mathcal{H}^B = \left(\mathbb{C}^{2N} \right)^{\otimes r+1}$. Let us focus on the space $\mathcal{H}^A$. Since $S^{(r)}$ is the reverse order operator, every computational basis component of the state $S^{(r)} \, | \Phi_k^{(r)\perp} \rangle$ has at least one subsystem $s \in [r+1]$ of the form $| k_s + N \rangle \in \mathbb{C}^{2N}$, while every computational basis component of $| \Phi_j^{(r)} \rangle$ or $| \Phi_j^{(r)\perp} \rangle$ has all subsystems of the form $| j_t \rangle$ for $t \in [r+1]$. The orthogonality statement immediately follows from $\langle j_t | k_s + N \rangle = 0$ for any $j_t, k_s \in [N]$.

□

Lemma 3.8. *$Q^{(r)} = T^{(r)\dagger} S^{(r)} T^{(r)}$ is a $(1, 2rn + 2r + n + 2, \epsilon)$-block-encoding of $(H/d)^r$, if $T^{(r)}$ is implemented to precision $\epsilon/2$.*

Proof. We first show

$$\left(\mathbb{1} \otimes \langle 0 |^{\otimes 2rn+2r+n+2} \right) T^{(r)\dagger} S^{(r)} T^{(r)} \left(\mathbb{1} \otimes | 0 \rangle^{\otimes 2rn+2r+n+2} \right) = (H/d)^r \tag{9}$$

for precise $T^{(r)}$. This part is similar to the proof of Lemma 3.3, and it suffices to show

$$\langle \Psi_j^{(r)} | \, S^{(r)} \, | \Psi_l^{(r)} \rangle = \langle j | \, (H/d)^r \, | l \rangle \tag{10}$$

for all $j, l \in [N]$, where $| \Psi_j^{(r)} \rangle$ is defined as above. Equation (10) can be obtained by splitting $| \Psi_j^{(r)} \rangle = | \Phi_j^{(r)} \rangle + | \Phi_j^{(r)\perp} \rangle$ and then applying Lemma 3.7.

For approximated $T^{(r)}$ with precision $\epsilon/2$, by linearity of error bound propagation, the LHS of (9) is approximated to precision ϵ. □

In order to implement an r-parallel quantum walk $Q^{(r)}$, we only need to focus on the $T^{(r)}$ part, since $S^{(r)}$ can be trivially implemented by a quantum circuit of constant depth using SWAP gates. The outline of an implementation of $T^{(r)}$ is presented in Figure 4.

Note that the implementation consists of two stages: pre-walk and re-weight, where the two oracles $\mathcal{O}_H$ and $\mathcal{O}_P$ are queried separately — $\mathcal{O}_P$ is only queried in the pre-walk stage, while $\mathcal{O}_H$ is only queried in the re-weight stage.

State space

$\mathcal{H} = \left(\bigotimes_{s\in[r+1]} \mathcal{H}_s^A\right) \otimes \left(\bigotimes_{s\in[r+1]} \mathcal{H}_s^B\right)$, where $\mathcal{H}_s^A = \mathcal{H}_s^B = \mathbb{C}^{2N}$ for all s.

Input

Any state $|j_0\rangle \otimes |0\rangle \otimes \ldots \otimes |0\rangle$ for $j_0 \in [N]$ (due to the definition of $T^{(r)}$ in (7)), with $|j_0\rangle, |0\rangle \in \mathbb{C}^{2N}$.

Output

The state $|\Psi_{j_0}^{(r)}\rangle$ defined in (6).
$T^{(r)}$ can be implemented in the following ways.

Pre-walk

1. Prepare in the subspace $\mathcal{H}^A$ a pre-walk state

$$|p_{j_0}^{(r)}\rangle := \frac{1}{\sqrt{d^r}} \sum_{\boldsymbol{j}\in \mathsf{H}^r} |\boldsymbol{j}\rangle \in \mathcal{H}^A. \tag{11}$$

Re-weight

2. Copy the computational basis states in $\mathcal{H}^A$ to $\mathcal{H}^B$; that is, apply $\mathrm{COPY}^{(r+1)\cdot(n+1)}$ to obtained the state

$$\frac{1}{\sqrt{d^r}} \sum_{\boldsymbol{j}\in \mathsf{H}^r} |\boldsymbol{j}\rangle |\boldsymbol{j}\rangle \in \mathcal{H}^A \otimes \mathcal{H}^B.$$

3. Query r copies of the oracle $\mathcal{O}_H$ in parallel, each in the subspace $\mathcal{H}_s^A \otimes \mathcal{H}_{s+1}^B$ for $s \in [r]$.[13] Similar to the proof of Lemma 3.4, for each query we compute $H_{j_s j_{s+1}}$ in a temporary ancilla space, conditioned on which rotates the state in $\mathcal{H}_{s+1}^B$, then uncompute $H_{j_s j_{s+1}}$. Finally we obtain the goal state $|\Psi_{j_0}^{(r)}\rangle$.

Figure 4: Implementation of $T^{(r)}$.

3.2.1 Pre-walk and Uniform-structured Hamiltonians

Now we give a detailed description of pre-walk. At the same time, we introduce a class of Hamiltonians — uniform-structured Hamiltonians, for which the pre-walk can be conducted in a parallel fashion. The state $|p_{j_0}^{(r)}\rangle = \frac{1}{\sqrt{d^r}} \sum_{\boldsymbol{j}\in \mathsf{H}^r} |\boldsymbol{j}\rangle$ in (11) earns the name "pre-walk state" because it is a superposition of all paths generated by r steps of *un-*

[13] Here we exploit the two copies of $|\boldsymbol{j}\rangle$ to parallelize the queries to the oracle $\mathcal{O}_H$. The idea of using multiple copies of data is intuitive and ubiquitous in classical parallel computing. An alternative way to achieve the parallelization is: given a single copy of $|\boldsymbol{j}\rangle \in \mathcal{H}^A$, first query $\lceil r/2 \rceil$ copies of $\mathcal{O}_H$ in parallel, each in the subspace $\mathcal{H}_s^A \otimes \mathcal{H}_{s+1}^A$ for even s; and then query $\lfloor r/2 \rfloor$ copies of $\mathcal{O}_H$ in parallel, each in $\mathcal{H}_s^A \otimes \mathcal{H}_{s+1}^A$ for odd s. This technique is used later for proving Corollary 3.19. In this alternative way, one can also modify Definition 3.5 (and relevant statements, including Lemma 3.7) such that (8) only contains one copy of $|\boldsymbol{j}\rangle$.

weighted random walk on the graph H starting from the vertex j_0. We call the process of generating $|p_{j_0}^{(r)}\rangle$ an *r-pre-walk*. For simplicity, we assume $|p_{j_0}^{(r)}\rangle \in \left(\mathbb{C}^N\right)^{\otimes r+1}$, that is, $|j_s\rangle \in \mathbb{C}^N$ for all $|j_s\rangle$.

For the pre-walk, we only need to focus on the graph H and the oracle $\mathcal{O}_P$ that characterizes its sparse structure, as $|p_{j_0}^{(r)}\rangle$ does not involve any weight H_{jk}. One remaining question is what kind of oracle $\mathcal{O}_P$ to be used in our algorithm. Since the complexities that we consider are measured in terms of the query complexity with respect to $\mathcal{O}_P$ and gate complexity, for practical reasons, $\mathcal{O}_P$ should be reasonably efficiently implementable. Conversely, if $\mathcal{O}_P$ is powerful enough (thus hard to implement), then intuitively the pre-walk can be done with only a few queries to $\mathcal{O}_P$. For instance, given an oracle $\mathcal{O}_P = \mathcal{O}_{path}$ that directly gives the path generated from walks according to a sequence of choices, as shown in the following lemma, then the query complexity is $O(1)$. Recall that $L(j,t)$ denotes the column index of the t^{th} nonzero entry in row j of H, i.e., the t^{th} neighbor of vertex j in the graph H.

Lemma 3.9 (Pre-walk with a strong path oracle $\mathcal{O}_{path}$)**.** *Let $\mathcal{O}_P = \mathcal{O}_{path}$ give a path generated by r steps of walk starting from j_0, according to the sequence of choices $\boldsymbol{t} \in [d]^r$; that is, take $\mathcal{X} = [N], \mathcal{Y} = [N]^r$, and $P(j_0, \boldsymbol{t}) = (j_1, \ldots, j_r)$ with $j_{s+1} := L(j_s, t_s)$ for $s \in [r]$. Then the r-pre-walk can be implemented by a quantum circuit of*

- *size $O(1)$ w.r.t. queries to $\mathcal{O}_P$, and*
- *depth $O(1)$ and size $O(r \log d)$ w.r.t. gates.*

Proof. Assume d is a power of two w.l.o.g. Starting from the initial state $|j_0\rangle |0\rangle^{\otimes r}$, the pre-walk can be implemented in the following way. First prepare a superposition $\frac{1}{\sqrt{d^r}} \sum_{\boldsymbol{t} \in [d]^r} |\boldsymbol{t}\rangle$ in the second register by an $O(1)$-depth and $O(r \log d)$-size quantum circuit using Hadamard gates, then query $\mathcal{O}_P$ to compute $P(j_0, \boldsymbol{t}) = (j_1, \ldots, j_r)$ and thus obtain the goal state $|p_{j_0}^{(r)}\rangle$. □

In general, it might be expensive to implement $\mathcal{O}_{path}$. For example, given the oracle $\mathcal{O}_L$ that computes the function L, the straightforward way to implement the strong oracle $\mathcal{O}_{path}$ requires r sequential queries to $\mathcal{O}_L$. However, for a special class of Hamiltonians, generating a path according to a sequence of choices can be done more efficiently by exploiting parallelism in computing compositions of the function L. This forms the basic idea of uniform-structured Hamiltonians. Let function $L^{(r)} : [N] \times [d]^r \to [N]$ be inductively defined as

$$L^{(r)}(j, t_0, \ldots, t_{r-1}) := L\Big(L^{(r-1)}(j, t_0, \ldots, t_{r-2}), t_{r-1}\Big)$$

for $j \in [N], \boldsymbol{t} \in [d]^r$, with $L^{(1)} := L$. Note that $L^{(r)}$ gives the destination of r steps of walk according to a sequence of choices. Uniform-structured Hamiltonians are a class of Hamiltonians for which the function $L^{(r)}$ can be computed efficiently in parallel.

Definition 3.10 (Uniform-structured Hamiltonian)**.** A d-sparse Hamiltonian H with the associated oracle $\mathcal{O}_P$ is uniform-structured if:

- For all $r \in \mathbb{N}$, the corresponding $L^{(r)}$ can be expressed as

$$L^{(r)}(j, \boldsymbol{t}) = f(j, g(t_0) \circ \ldots \circ g(t_{r-1})) \tag{12}$$

  where the function f, g and the operator $\circ$ with input/output lengths $O(n)$ satisfy that:

– f and the mapping $|0\rangle \mapsto \frac{1}{\sqrt{d}} \sum_{t\in[d]} |g(t)\rangle$ are arithmetic-depth-efficiently computable (see Definition 2.7) with $O(1)$ queries to $\mathcal{O}_P$;

– $\circ$ is associative and arithmetic-depth-efficiently computable.

- There exists an "inverse" function $L^{(-1)}$ such that $L^{(-1)}(j, L(j,t)) = g(t)$ for all $j \in [N], t \in [d]$, and $L^{(-1)}$ is arithmetic-depth-efficiently computable with $O(1)$ queries to $\mathcal{O}_P$.

Remark 3.11. One might notice in the first condition that the expression $g(t_0) \circ \ldots \circ g(t_{r-1})$ is ready to be computed in parallel by Lemma 2.3, which is actually the key to implement a parallel pre-walk. We also point out that if the function g and its inverse are both arithmetic-depth-efficiently computable, then the mapping $|0\rangle \mapsto \frac{1}{\sqrt{d}} \sum_{t\in[d]} |g(t)\rangle$ is also arithmetic-depth-efficiently computable by evaluating g in place on the state $\frac{1}{\sqrt{d}} \sum_{t\in[d]} |t\rangle$, which can be prepared by applying Hadamard gates in parallel.

In the second condition, the inverse function $L^{(-1)}$ actually enables an efficient garbage cleaning, as shown later in Corollary 3.19.

For a better understanding of the quite involved Definition 3.10, we show three examples of uniform-structured Hamiltonians. The first example is a band Hamiltonian, which has its nonzero entries concentrated within a band around the diagonal.

Lemma 3.12 (Band Hamiltonian)**.** *Assume $d \in [N]$ is odd. Let H be a d-band Hamiltonian, i.e., $H_{jk} = 0$ if $k \notin \mathcal{B}_j^d$, where $\mathcal{B}_j^d := \{j + t - (d-1)/2 : t \in [d]\}$ with the addition and subtraction in $\mathbb{Z}_N$.*[14] *Let $\mathcal{O}_P$ be an empty oracle, that is, take $\mathcal{X} = \mathcal{Y} = \emptyset$ and P to be undefined. Then H is uniform-structured.*

Example 3.13. The 4×4 Hamiltonian H with matrix form

$$H := \begin{bmatrix} 1 & i & 0 & 0 \\ -i & 2 & 3 & 0 \\ 0 & 3 & -1 & -i \\ 0 & 0 & i & 1 \end{bmatrix}$$

is 3-band.

Proof of Lemma 3.12. Note that for band Hamiltonians we do not need to query $\mathcal{O}_P$. The lemma is proved by verifying the conditions in Definition 3.10. Note that $L(j,t) = j + t - (d-1)/2$, where the addition and subtraction are in $\mathbb{Z}_N$ (although in this case the nonzero entries are not necessarily ordered, the correctness of the algorithm is unaffected).

- We have

$$L^{(r)}(j, \boldsymbol{t}) = j + (t_0 - (d-1)/2) + \ldots + (t_{r-1} - (d-1)/2);$$

that is, take $f(j,k) = j + k$, $g(t) = t - (d-1)/2$, and $\circ$ to be addition in (12).

– By Lemma 2.6, f is arithmetic-depth-efficiently computable. As in Remark 3.11, the mapping $|0\rangle \mapsto \frac{1}{\sqrt{d}} \sum_{t\in[d]} |g(t)\rangle$ is arithmetic-depth-efficiently computable too.

[14] This allows a slightly more general definition than the usual band Hamiltonian, because in $\mathbb{Z}_N$ when the band has its center close to 0 it will wrap back from the $N-1$ side. For instance, taking $H_{03} = H_{30} = 1$ in Example 3.13 H is still 3-band.

– The addition is obviously associative; and is arithmetic-depth-efficiently computable by Lemma 2.6.

- Take the inverse function to be $L^{(-1)}(j,k) = k - j$, which is arithmetic-depth-efficiently computable by Lemma 2.6.

□

The second example is a tensor product of Pauli matrices. Recall that any Hamiltonian can be expressed as a sum of (scaled) tensor products of Pauli matrices, which form a basis for the Hermitian space. In Section 3.3, we will further show that this Pauli sum is also uniform-structured (according to the extended definition).

Lemma 3.14 (Tensor product of Pauli matrices)**.** *Let H be a (scaled) tensor product of Pauli matrices, that is, $H = \alpha \bigotimes_{k\in[n]} \sigma_k$ with $\sigma_k \in \{\mathbb{1}, X, Y, Z\}$ and α being a constant. Let $\mathcal{O}_P$ give an n-bit string s characterizing the Pauli string $\langle \sigma_k \rangle_{k\in[n]}$. In particular, take $\mathcal{X} = [1]$, $\mathcal{Y} = [N]$ and $P(x,y) = y \oplus s$, where the k^{th} bit of s is defined as*

$$s_k := \begin{cases} 0, & \sigma_k \in \{\mathbb{1}, Z\} \text{ (diagonal)}, \\ 1, & \sigma_k \in \{X, Y\} \text{ (off-diagonal)}. \end{cases}$$

Then H is uniform-structured.

Proof. Note that the sparsity $d = 1$ for H, because all Pauli matrices are 1-sparse. The lemma is proved by verifying the conditions in Definition 3.10. It holds that $L(j,t) = j \oplus s$ where $\oplus$ is the bit-wise XOR operator.

- We have
$$L^{(r)}(j, \boldsymbol{t}) = j \oplus s \oplus \ldots \oplus s;$$
that is, take $f(j,k) = j \oplus k$, $g(t) = s$, and $\circ$ to be $\oplus$ in (12).
  - f is arithmetic-depth-efficiently computable, while the mapping $|0\rangle \mapsto \frac{1}{\sqrt{d}} \sum_{t\in[d]} |g(t)\rangle$ is trivial to perform with a single query to $\mathcal{O}_P$ because $d = 1$.
  - The XOR $\oplus$ is obviously associative and arithmetic-depth-efficiently computable.
- Take the inverse function to be $L^{(-1)}(j,k) = s$, which can be computed by a single query to $\mathcal{O}_P$.

□

The third example is a local Hamiltonian term, which acts non-trivially on a subsystem of l qubits, whose positions are indicated by the l bits of 1 in an n-bit string s. The sum of many local Hamiltonian terms is a local Hamiltonian, which will be investigated in Section 3.3 as a uniform-structured Hamiltonian (according to the extended definition).

Lemma 3.15 (Local Hamiltonian term)**.** *Let H be an l-local Hamiltonian term; that is, $H = H_s \otimes \mathbb{1}_{\bar{s}}$, where H_s is a Hamiltonian acting on the subsystem of l qubits whose positions are indicated by l bits of 1 in the n-bit string s, and $\mathbb{1}_{\bar{s}}$ is the identity operator on the subsystem of the rest $n - l$ qubits. Let $\mathcal{O}_P$ give the locality parameter s. In particular, set $\mathcal{X} = [1]$, $\mathcal{Y} = [N]$ and $P(x,y) = y \oplus s$. Then H is uniform-structured.*

Example 3.16. Let $H := A \otimes \mathbb{1} \otimes B \otimes \mathbb{1}$ be a 16×16 Hamiltonian with A and B being 2×2 Hamiltonians, and $\mathbb{1}$ being 2×2 identity matrix. Then H is a 2-local Hamiltonian term $H_s \otimes \mathbb{1}_{\bar{s}}$, with $H_s = A \otimes B$ and $s = 1010$.

Proof of Lemma 3.15. The lemma is proved by verifying the conditions in Definition 3.10. We use the superscript i to denote the i^{th} bit of a number. Note that the sparsity $d = 2^l$, $L(j,t) = j \lhd_s (t \restriction_s)$, where the operator $\restriction_s \colon [d] \to [N]$ lifts an l-bit string to an n-bit string according to s, defined as

$$(b \restriction_s)^i := s^i \cdot b^{s^0+\ldots+s^i},$$

and the operator $\lhd_s \colon [N] \times [N] \to [N]$ overwrites an n-bit string by another according to s, defined as

$$(a \lhd_s b)^i := a^i(1-s^i) + b^i s^i \tag{13}$$

for all $a, b \in [N], i \in [n]$. For instance, $101 \restriction_{01011} = 01001$ and $10011 \lhd_{01011} 01001 = 11001$.

Technically, we further define two operators $\lhd \colon [N] \times [N]^2 \to [N]$ and $\lhd^\vee \colon [N]^2 \times [N]^2 \to [N]^2$ such that

$$a \lhd (b,x) := a \lhd_x b, \qquad (a,x) \lhd^\vee (b,y) := (a \lhd_y b, x \vee y) \tag{14}$$

for all $a, b, x, y \in [N], i \in [n]$, where $\vee$ is the bit-wise OR operator.

- We have

$$L^{(r)}(j, \boldsymbol{t}) = j \lhd (t_0 \restriction_s, s) \lhd^\vee \ldots \lhd^\vee (t_{r-1} \restriction_s, s);$$

  that is, take $f(j,(k,x)) = j \lhd (k,x)$, $g(t) = (t \restriction_s, s)$, and $\circ$ to be $\lhd^\vee$ in (12).

  - To compute f, it suffices to compute the operator $\lhd$, which according to (13) and (14) is arithmetic-depth-efficiently computable.
    The mapping $|0\rangle \mapsto \frac{1}{\sqrt{d}} \sum_{t \in [d]} |g(t)\rangle$ can be performed by first querying $\mathcal{O}_P$ to obtain $|s\rangle$ in an ancilla, then conditioned on it implementing n controlled Hadamard gates in parallel, assuming d is a power of two w.l.o.g. This is arithmetic-depth-efficient with $O(1)$ queries.
  - The associativity of $\lhd^\vee$ is easy to verify by noting that $a \lhd_x b \lhd_y c = a \lhd_x (b \lhd_y c)$ for all $a, b, c, x, y \in [N]$ and the associativity of $\vee$. Also, $\lhd^\vee$ is arithmetic-depth-efficiently computable by (14).

- Take the inverse function to be $L^{(-1)}(j,k) = (k \wedge s, s)$, where $\wedge$ is the bit-wise AND operator. This is arithmetic-depth-efficiently computable with $O(1)$ queries to $\mathcal{O}_P$.

□

Now we are ready to present the parallel pre-walk subroutine for uniform-structured Hamiltonians. As aforementioned, the parallelism relies on the structure of $L^{(r)}$, which will be shown to be computable by a logarithmic depth quantum circuit with respect to r. The goal is achieved through several lemmas and corollaries.

Lemma 3.17. *For a uniform-structured Hamiltonian H, the mapping*

$$|j\rangle\,|g(\boldsymbol{t})\rangle\,|z\rangle \mapsto |j\rangle\,|g(\boldsymbol{t})\rangle\,|z \oplus L^{(r)}(j,\boldsymbol{t})\rangle$$

for all $j, z \in [N], \boldsymbol{t} \in [d]^r$ with $g(\boldsymbol{t}) = (g(t_0), \ldots, g(t_{r-1}))$, can be implemented by a quantum circuit of

- *size $O(1)$ w.r.t. queries to $\mathcal{O}_P$, and*
- *depth $O\left(\log r \cdot \log^2 n\right)$ and size $O(rn^4)$ w.r.t. gates.*

Proof. Since $\circ$ is associative and arithmetic-depth-efficiently computable, by Lemma 2.3 we can first evaluate $g(t_0) \circ \ldots \circ g(t_{r-1})$ by an $O\Big(\log r \cdot \log^2 n\Big)$-depth and $O(rn^4)$-size quantum circuit. Also f is arithmetic-depth-efficiently computable with $O(1)$ queries to $\mathcal{O}_P$, thus $L^{(r)}(j, \boldsymbol{t}) = f(j, g(t_0) \circ \ldots \circ g(t_{r-1}))$ can be computed by additional depth $O\Big(\log^2 n\Big)$ and size $O(n^4)$ w.r.t. gates and $O(1)$ queries. Finally apply COPY^n to copy the result into $|z\rangle$, followed by garbage cleaning using reverse computation. The total complexity follows from summing these complexities up. □

Note that for different $s \in [r+1]$, the function $L^{(s)}$ can be computed in parallel, then we obtain the following corollary.

Corollary 3.18. *For a uniform-structured Hamiltonian H, the mapping*

$$|j\rangle\,|g(\boldsymbol{t})\rangle\,|\boldsymbol{z}\rangle \mapsto |j\rangle\,|g(\boldsymbol{t})\rangle \bigotimes_{s=1}^{r} |z_{s-1} \oplus L^{(s)}(j, t_0, \ldots, t_{s-1})\rangle \tag{15}$$

for all $j \in [N], \boldsymbol{t} \in [d]^r, \boldsymbol{z} \in [N]^r$, can be implemented by a quantum circuit of

- *depth $O(1)$ and size $O(r)$ w.r.t. queries to $\mathcal{O}_P$, and*
- *depth $O\Big(\log r \cdot \log^2 n\Big)$ and size $O(r^2 n^4)$ w.r.t. gates.*

Proof. First make r copies of $|j\rangle\,|g(\boldsymbol{t})\rangle$ by applying parallel $\mathrm{COPY}_r^{O(rn)}$ in Lemma 2.1, which has depth $O(\log r)$ and size $O(r^2 n)$. Then apply Lemma 3.17 to each of the r copies, for the s^{th} copy taking the input $(j, g(t_0), \ldots, g(t_{s-1}))$ to compute $L^{(s)}(j, t_0, \ldots, t_{s-1})$, which can be done in parallel for $s = 1$ to r. Finally apply COPY^n in parallel to store the result, followed by garbage cleaning using reverse computation. The final complexity comes from summing up these complexities. □

With the help of the inverse function $L^{(-1)}$, we can generate a path state $|\boldsymbol{j}\rangle$ from the state $|j_0, g(\boldsymbol{t})\rangle$ by erasing the redundant information $|g(\boldsymbol{t})\rangle$ in (15), as shown in the following corollary.

Corollary 3.19. *For a uniform-structured Hamiltonian H, the mapping*

$$|j_0, g(\boldsymbol{t})\rangle \mapsto |\boldsymbol{j}\rangle$$

for all $j_0 \in [N], \boldsymbol{t} \in [d]^r$, where $\boldsymbol{j} \in \mathsf{H}^r$ has $j_{s+1} := L(j_s, t_s)$ for $s \in [r]$, can be implemented by a quantum circuit of

- *depth $O(1)$ and size $O(r)$ w.r.t. queries to $\mathcal{O}_P$, and*
- *depth $O\Big(\log r \cdot \log^2 n\Big)$ and size $O(r^2 n^4)$ w.r.t. gates.*

Proof. Since $j_s = L^{(s)}(j_0, t_0, \ldots, t_{s-1})$ for $s = 1$ to r, to perform the required mapping we only need to clean the $|g(\boldsymbol{t})\rangle$ in (15) after applying Corollary 3.18 to $|j_0, g(\boldsymbol{t})\rangle\,|0\rangle^{\otimes r}$. Recall that $L^{(-1)}(j_s, j_{s+1}) = g(t_s)$ for $s \in [r]$, so we can compute $L^{(-1)}$ by taking inputs $|j_s, j_{s+1}\rangle$ to clean $g(t_s)$ first on odd s then on even s, thereby for different s the computation can be done in parallel. Since $L^{(-1)}$ is arithmetic-depth-efficiently computable with $O(1)$ queries to $\mathcal{O}_P$, the cleaning process can be done by a quantum circuit of depth $O\Big(\log^2 n\Big)$ and size $O(rn^4)$ w.r.t. gates and $O(1)$ queries. The final complexity follows from summing these complexities up. □

A combination of the above results gives a parallel pre-walk algorithm for uniform-structured Hamiltonians.

Lemma 3.20 (Pre-walk on uniform-structured Hamiltonians)**.** *Let H be a uniform-structured Hamiltonian, then an r-pre-walk on its graph H, i.e., preparing the state $|p_{j_0}^{(r)}\rangle$, can be implemented by a quantum circuit of*

- *depth $O(1)$ and size $O(r)$ w.r.t. queries to $\mathcal{O}_P$, and*
- *depth $O\Big(\log r \cdot \log^2 n\Big)$ and size $O(r^2 n^4)$ w.r.t. gates.*

Proof. Let $\bigotimes_{s\in[r+1]} \mathcal{H}_s$ be the state space with $\mathcal{H}_s = \mathbb{C}^N$. The process of preparing $|p_{j_0}^{(r)}\rangle$ is shown below, starting from the initial state $|j_0\rangle |0\rangle^{\otimes r}$ with $|j_0\rangle, |0\rangle \in \mathbb{C}^N$.

1. Perform the mapping $|0\rangle \mapsto \frac{1}{\sqrt{d}} \sum_{t\in[d]} |g(t)\rangle$ in each $\mathcal{H}_s$ for $s = 1$ to r, to obtain the state
$$\frac{1}{\sqrt{d^r}} |j_0\rangle \sum_{\boldsymbol{t}\in[d]^r} |g(\boldsymbol{t})\rangle .$$
2. Apply Corollary 3.19 we obtain the goal state $|p_{j_0}^{(r)}\rangle = \frac{1}{\sqrt{d^r}} \sum_{\boldsymbol{j}\in\mathsf{H}^r} |\boldsymbol{j}\rangle$.

Each mapping $|0\rangle \mapsto \frac{1}{\sqrt{d}} \sum_{t\in[d]} |g(t)\rangle$ in Step 1 is arithmetic-depth-efficient with $O(1)$ queries to $\mathcal{O}_P$, due to the definition of uniform-structured Hamiltonians. Combined with Corollary 3.19 the final complexity is obtained. □

3.2.2 Re-weight

Intuitively, the re-weight procedure in the implementation of $T^{(r)}$ in Figure 4 adjusts the "weight" of each path $|\boldsymbol{j}\rangle$ in the pre-walk state $|p_{j_0}^{(r)}\rangle$ according to the entries in H, given by the oracle $\mathcal{O}_H$. As we will see in the following, the re-weight analysis is in fact simpler than the pre-walk because there is no requirement on the sparse structure of H.

Lemma 3.21 (Re-weight)**.** *Re-weight of $|p_{j_0}^{(r)}\rangle$, i.e., performing the mapping $|p_{j_0}^{(r)}\rangle \mapsto |\Psi_{j_0}^{(r)}\rangle$, where $|\Psi_{j_0}^{(r)}\rangle$ is defined in (6), can be implemented to precision ϵ by a quantum circuit of*

- *depth $O(1)$ and size $O(r)$ w.r.t. queries to $\mathcal{O}_H^b$ with $b = O(\log(1/\epsilon))$, and*
- *depth $O\Big(\log^2 \log(1/\epsilon)\Big)$ and size $O\Big(r\Big[n + \log^4(1/\epsilon)\Big]\Big)$ w.r.t. gates,*

for $r = \mathrm{polylog}(1/\epsilon)$.

Proof. We analyze separately the query complexity and gate complexity of the re-weight stage, including Step 2 and Step 3 in Figure 4.

- For query complexity, only Step 3 involves queries to $\mathcal{O}_H$. As $\mathcal{H}_s^A \otimes \mathcal{H}_{s+1}^B$ are disjoint for $s \in [r]$, these r queries are independent, and thus can be done in parallel with depth $O(1)$ and size $O(r)$. The precision b of oracle $\mathcal{O}_H^b$ will be determined in the gate complexity analysis below.

- For gate complexity, in Step 2 the $\mathrm{COPY}^{(r+1)\cdot(n+1)}$ gate can be implemented by an $O(1)$-depth and $O(rn)$-size quantum circuit. In Step 3, one needs to apply r controlled rotations conditioned on some H_{jk}; that is, perform the mapping

$$|0\rangle |H_{jk}\rangle \mapsto \left(\sqrt{H_{jk}^*}\,|0\rangle + \sqrt{1-|H_{jk}|}\,|1\rangle\right)|H_{jk}\rangle\,. \tag{16}$$

  To achieve a total precision ϵ of $T^{(r)}$, each rotation in (16) needs to be $O(\epsilon/r)$-precise, which requires H_{jk} given by the oracle $\mathcal{O}_H^b$ to have $b = O(\log(r/\epsilon)) = O(\log(1/\epsilon))$ bits of precision. To perform the rotation, first compute $\sqrt{(1-|H_{jk}|)/H_{jk}^*}$ and its arctangent arithmetic-depth-efficiently by Lemma 2.6, then apply C_b-R_Y in Corollary 2.2.

  Following [9], to satisfy the condition $\sqrt{H_{jk}^*}(\sqrt{H_{jk}})^* = H_{jk}^*$ one should be careful in choosing the sign of $\sqrt{H_{jk}^*}$ for $H_{jk} < 0$. This problem is addressed by adding an $O\left(2^{-b}\right)$ disturbance on the imaginary part of H_{jk} to force it to be nonzero (thereby forcing H_{jk} to be complex) for those $H_{jk} < 0$, with the total precision unchanged up to a constant factor.[15] Now for $H_{jk} = re^{i\theta}$, let $\sqrt{H_{jk}^*} := \sqrt{r}e^{i\theta/2}$, which is also arithmetic-depth-efficiently computable by Lemma 2.6. From Corollary 2.2, a C_b-R_Y gate can be implemented by an $O(\log b)$-depth and $O(b)$-size quantum circuit. The total complexity follows by summing up the complexities of r rotations.

□

Finally, combining the pre-walk complexity (Lemma 3.20 in Section 3.2.1), the re-weight complexity (Lemma 3.21 in the above), and the negligible complexity of implementing $S^{(r)}$ gives the total complexity of the parallel quantum walk for uniform-structured Hamiltonians.

Theorem 3.22 (Parallel quantum walks for uniform-structured Hamiltonians)**.** *For a uniform-structured Hamiltonian H, the r-parallel quantum walk $Q^{(r)}$ can be performed to precision ϵ by a quantum circuit of*

- *depth $O(1)$ and size $O(r)$ w.r.t. queries to $\mathcal{O}_H^b$ with $b = O(\log(1/\epsilon))$,*
- *depth $O(1)$ and size $O(r)$ w.r.t. queries to $\mathcal{O}_P$, and*
- *depth $O\left(\log r \cdot \log^2 n + \log^2\log(1/\epsilon)\right)$ and size $O\left(r^2n^4 + r\log^4(1/\epsilon)\right)$ w.r.t. gates,*

for $r = \mathrm{polylog}(1/\epsilon)$.

3.3 Extension: A Parallel Quantum Walk for a Sum of Hamiltonians

As shown in the previous section, the parallel quantum walk can be efficiently implemented in parallel for the class of uniform-structured Hamiltonians, which are however somewhat restricted in applications. Now we extend the framework in Section 3.2 to a parallel quantum walk for *a sum of Hamiltonians*, that is, a Hamiltonian of the form $H = \sum_{w\in[m]} H_w$,

[15] In [9], to avoid this sign ambiguity, they assign different signs to $\sqrt{H_{jk}^*}$ for H_{jk} above and below the diagonal, and replace H with $H + \|H\|_{\max}\mathbb{1}$ to force the diagonal elements to be non-negative. For Hamiltonian simulation problem, this modification works well by only introducing a global phase. Here we provide an alternative solution that is not restricted to the Hamiltonian simulation task.

where H_w are d-sparse Hamiltonians and $m = \mathrm{poly}(n)$. In this extended framework, we generalize the class of uniform-structured Hamiltonians to include more Hamiltonians of practical interest, like Pauli sums and local Hamiltonians.

Recall that in Section 3.2 the good sparse structure of a Hamiltonian H is a key to efficiently implement the parallel quantum walk. The intuition behind the extended framework in this section is then: if some Hamiltonians has the same type of good sparse structures, then a sum of them is still structured well enough for exploiting parallelism. The organization of this section is similar to Section 3.2. For readability, we only provide the essential definitions and lemmas here, and leave more details to Appendix A.

Let us first define the extended parallel quantum walk for a sum of Hamiltonians. Recall that the state $|\Psi_{j_0}^{(r)}\rangle$ in (6), which is a superposition of paths $\boldsymbol{j} = (j_0, \ldots, j_{r-1})$ in the graph H, is a key ingredient in the parallel quantum walk in the previous section. Now in the case of a sum Hamiltonian $H = \sum_w H_w$, the graph H can be "decomposed" into a sum of subgraphs H_w, so in a path $\boldsymbol{j} \in \mathsf{H}$ each edge (j_s, j_{s+1}) belongs to at least one subgraph. Thus, we can define an extended state $|\Psi_{j_0}^{(r,m)}\rangle$, which is still a superposition of paths $\boldsymbol{j} \in \mathsf{H}$, but each tensored with a corresponding string $\boldsymbol{w}$, such that $(j_s, j_{s+1}) \in H_{w_s}$ for all $s \in [r]$. In this way, the extended parallel quantum walk can better exploit the sum structure of the Hamiltonian $H = \sum_{w\in[m]} H_w$, as shown later.

Definition 3.23 ((r,m)-parallel quantum walk)**.** Given the Hamiltonian H as above. Let $\mathcal{H} = \mathcal{H}^W \otimes \mathcal{H}^A \otimes \mathcal{H}^B$ be the walk space, where $\mathcal{H}^W = (\mathbb{C}^m)^{\otimes r}$ and $\mathcal{H}^A = \mathcal{H}^B = \left(\mathbb{C}^{2N}\right)^{\otimes r+1}$. For each $j_0 \in [N]$, define $|\Psi_{j_0}^{(r,m)}\rangle :=$

$$\frac{1}{\sqrt{(md)^r}} \sum_{\boldsymbol{w}\in[m]^r} \sum_{\boldsymbol{j}\in\mathsf{H}^{\boldsymbol{w}}} \underbrace{|\boldsymbol{w}\rangle}_{\in\mathcal{H}^W} \otimes \underbrace{|\boldsymbol{j}\rangle}_{\in\mathcal{H}^A} \otimes |j_0\rangle \underbrace{\bigotimes_{s\in[r]} \left(\sqrt{\tilde{H}^*_{j_s j_{s+1}}}\, |j_{s+1}\rangle + \sqrt{1 - \left|\tilde{H}_{j_s j_{s+1}}\right|}\, |j_{s+1} + N\rangle \right)}_{\in\mathcal{H}^B} \tag{17}$$

where $\boldsymbol{j} \in \mathsf{H}^{\boldsymbol{w}}$ denotes $(j_s, j_{s+1}) \in \mathsf{H}_{w_s}$ for all $s \in [r]$, and $\tilde{H}_{jk} := H_{jk}/c(j,k)$ with $c(j,k) := \sum_{w\in[m]} [\![(j,k) \in \mathsf{H}_w]\!]$[16] the number of subgraphs containing the edge $(j,k) \in \mathsf{H}$. Let

- $T^{(r,m)} : \mathcal{H} \to \mathcal{H}$ be any unitary operator such that

$$T^{(r,m)}(|\boldsymbol{w}\rangle\,|j\rangle\,|\boldsymbol{z}\rangle) = \begin{cases} |\Psi_j^{(r,m)}\rangle\,, & j \in [N], \boldsymbol{w} = 0, \boldsymbol{z} = 0, \\ \text{any state}, & o.w. \end{cases}$$

  for all $\boldsymbol{w} \in [m]^r, j \in [2N], \boldsymbol{z} \in [2N]^{2r+1}$.

- $S^{(r,m)} : \mathcal{H} \to \mathcal{H}$ reverses the order in the subspace $\mathcal{H}^A \otimes \mathcal{H}^B$; that is, $S^{(r,m)} = \mathbb{1}^W \otimes S^{(r)}$, where $S^{(r)} : \mathcal{H}^A \otimes \mathcal{H}^B \to \mathcal{H}^A \otimes \mathcal{H}^B$ is the reverse order operator in Definition 3.5.

Then a step of (r,m)-parallel quantum walk for H is defined as $Q^{(r,m)} := T^{(r,m)\dagger} S^{(r,m)} T^{(r,m)}$.

Remark 3.24. Note that in (17), the amplitudes are determined by a new Hamiltonian $\tilde{H}$, which is entry-wise rescaled from H. The rescaling factor for each entry H_{jk} is $c(j,k)$, i.e., the number of overlapping subgraphs H_w on the edge $(j,k) \in \mathsf{H}$. For later implementation of the extended parallel quantum walk, we will consider a new oracle $\mathcal{O}_{\tilde{H}}$ giving an entry of

[16] Here $[\![\cdot]\!]$ stands for the Iverson bracket; that is, $[\![p]\!] = 1$ if p is true and $[\![p]\!] = 0$ if p is false.

$\tilde{H}$ such that $\mathcal{O}_{\tilde{H}}|j,k,z\rangle = |j,k,z \oplus \tilde{H}_{jk}\rangle$. Note that if $c(j,k)$ can be efficiently computed, $\mathcal{O}_{\tilde{H}}$ can be easily constructed from $\mathcal{O}_H$, with the total precision scaled by at most m for the construction, as $c(j,k) = \sum_{w\in[m]}[\![(j,k) \in H_w]\!] \leq m$. The overhead caused by this precision scaling will be shown to be negligible later.

The extended parallel quantum walk defined above also block-encodes a monomial of H, as shown in the following lemma.

Lemma 3.25. *$Q^{(r,m)} = T^{(r,m)\dagger} S^{(r,m)} T^{(r,m)}$ is a $(1, r\lceil \log m \rceil + 2rn + 2r + n + 2, \epsilon)$-block-encoding of $\left(\frac{H}{md}\right)^r$, if $T^{(r,m)}$ is implemented to precision $\epsilon/2$.*

Proof. Postponed to Appendix A. □

As a straightforward generalization of the implementation of $T^{(r)}$ in Figure 4, an implementation of $T^{(r,m)}$ is shown in Figure 5. We call the procedure of preparing the (extended) pre-walk state $|p_{j_0}^{(r,m)}\rangle$ in (18) an (r,m)-pre-walk. Now we will introduce the notion of m-uniform-structured Hamiltonians as an extension of the uniform-structured Hamiltonians in Section 3.2. We first redefine the function $L : [m] \times [N] \times [d] \to [N]$ such that $L(w,j,t)$ is the t^{th} neighbor of vertex j in subgraph H_w, and let $L^{(r)} : [m]^r \times [N] \times [d]^r \to [N]$ be inductively defined as

$$L^{(r)}(\boldsymbol{w}, j, \boldsymbol{t}) := L\Big(w_{r-1}, L^{(r-1)}(w_0, \ldots, w_{r-2}, j, t_0, \ldots, t_{r-2}), t_{r-1}\Big)$$

for $\boldsymbol{w} \in [m]^r, j \in [N], \boldsymbol{t} \in [d]^r$, with $L^{(1)} := L$.

Definition 3.26 (m-uniform-structured Hamiltonian)**.** A sum Hamiltonian $H = \sum_{w\in[m]} H_w$ with the associated oracle $\mathcal{O}_P$ is m-uniform-structured if:

- For all $r \in \mathbb{N}$, its corresponding $L^{(r)}$ can be expressed as

$$L^{(r)}(\boldsymbol{w}, j, \boldsymbol{t}) = f(j, g(w_0, t_0) \circ \ldots \circ g(w_{r-1}, t_{r-1})) \tag{19}$$

  where the function f, g and the operator $\circ$ with input/output lengths $O(n)$ satisfy that:

  - f and the mapping $|w\rangle|0\rangle \mapsto |w\rangle \frac{1}{\sqrt{d}} \sum_{t\in[d]} |g(w,t)\rangle$ are arithmetic-depth-efficiently computable with $O(1)$ queries to $\mathcal{O}_P$ for all $w \in [m]$;
  - $\circ$ is associative and arithmetic-depth-efficiently computable.

- There exists an inverse function $L^{(-1)}$ such that $L^{(-1)}(w, j, L(w,j,t)) = g(w,t)$ for all $w \in [m], j \in [N], t \in [d]$, and $L^{(-1)}$ is arithmetic-depth-efficiently computable with $O(1)$ queries to $\mathcal{O}_P$.

- The function $[\![(j,k) \in \mathsf{H}_w]\!]$ can be arithmetic-depth-efficiently computed with $O(1)$ queries to $\mathcal{O}_P$, given $j, k \in [N], w \in [m]$ as inputs.

Note that the first two conditions in Definition 3.26 are naturally generalized from Definition 3.10, while the third condition is set to guarantee the efficiency of computing $c(j,k) = \sum_{w\in[m]}[\![(j,k) \in H_w]\!]$, which is used in the construction of the oracle $\mathcal{O}_{\tilde{H}}$ from $\mathcal{O}_H$, as mentioned in Remark 3.24.

Now we present two important examples of m-uniform-structured Hamiltonians: Pauli sums and local Hamiltonians, which are generalizations of tensor products of Pauli matrices and local Hamiltonian terms respectively.

State space

$\mathcal{H} = \left(\bigotimes_{s\in[r]} \mathcal{H}_s^W\right) \otimes \left(\bigotimes_{s\in[r+1]} \mathcal{H}_s^A\right) \otimes \left(\bigotimes_{s\in[r+1]} \mathcal{H}_s^B\right)$ with $\mathcal{H}_s^W = \mathbb{C}^m$ and $\mathcal{H}_s^A = \mathcal{H}_s^B = \mathbb{C}^{2N}$ for all s.

Input

Any state $|0\rangle^{\otimes mr} |j_0\rangle |0\rangle^{\otimes 2N(2r+1)}$ for $j_0 \in [N]$, with $|j_0\rangle \in \mathbb{C}^{2N}, |0\rangle \in \mathbb{C}$.

Output

The state $|\Psi_{j_0}^{(r,m)}\rangle$ defined in (17).
$T^{(r,m)}$ can be implemented in the following ways.

Pre-walk

1. Prepare in the subspace $\mathcal{H}^W \otimes \mathcal{H}^A$ a pre-walk state

$$|p_{j_0}^{(r,m)}\rangle := \frac{1}{\sqrt{(md)^r}} \sum_{\boldsymbol{w}\in[m]^r} \sum_{\boldsymbol{j}\in \mathsf{H}^{\boldsymbol{w}}} |\boldsymbol{w}\rangle |\boldsymbol{j}\rangle \in \mathcal{H}^W \otimes \mathcal{H}^A. \tag{18}$$

Re-weight

2. Copy the computational basis states in $\mathcal{H}^A$ to $\mathcal{H}^B$; that is, apply $\mathrm{COPY}^{(r+1)\cdot(n+1)}$ to obtained the state

$$\frac{1}{\sqrt{(md)^r}} \sum_{\boldsymbol{w}\in[m]^r} \sum_{\boldsymbol{j}\in \mathsf{H}^{\boldsymbol{w}}} |\boldsymbol{w}\rangle |\boldsymbol{j}\rangle |\boldsymbol{j}\rangle \in \mathcal{H}.$$

3. Query r copies of the modified oracle $\mathcal{O}_{\tilde{H}}$ in parallel, each in the subspace $\mathcal{H}_s^A \otimes \mathcal{H}_{s+1}^B$ for $s \in [r]$. For each query we compute $\tilde{H}_{j_s j_{s+1}}$ in a temporary ancilla space, conditioned on which rotates the state in $\mathcal{H}_{s+1}^B$, then uncompute $\tilde{H}_{j_s j_{s+1}}$ with another query. Finally we obtain the goal state $|\Psi_{j_0}^{(r,m)}\rangle$.

Figure 5: Implementation of $T^{(r,m)}$

Lemma 3.27 (Pauli sum)**.** *Let H be a Pauli sum, that is, $H = \sum_{w\in[m]} H_w$ where H_w are (scaled) tensor products of Pauli matrices defined in Lemma 3.14. Let $\mathcal{O}_P$ give an n-bit string $s(w)$ characterizing the Pauli string of H_w for each w. In particular, set $\mathcal{X} = [m]$, $\mathcal{Y} = [N]$ and $P(w, y) = y \oplus s(w)$. Then H is m-uniform-structured.*

Proof. Postponed to Appendix A. □

Lemma 3.28 (Local Hamiltonian)**.** *Let H be an (l, m)-local Hamiltonian, that is, $H = \sum_{w\in[m]} H_w$ where H_w are l-local Hamiltonian terms defined in Lemma 3.28. Let $\mathcal{O}_P$ give an n-bit string $s(w)$ characterizing the locality of H_w, i.e., the positions of the l qubits H_w acts on, for each w. In particular, set $\mathcal{X} = [m]$, $\mathcal{Y} = [N]$ and $P(w, y) = y \oplus s(w)$. Then H is m-uniform-structured.*

Proof. Postponed to Appendix A. □

Remark 3.29. Although any Hamiltonian can be represented as a Pauli sum due to the fact that tensor products of Pauli matrices form a basis for the Hermitian space, the number of summands m can be large. Since the complexity of the algorithm depends on m, only those Pauli sums with small m are of practical interest. The same difficulty exists when one tries to represent any Hamiltonian as a local Hamiltonian, because the parameter l can be large.

Following the same line of analysis as in Section 3.2, we have the following theorem that the (extended) parallel quantum walk for uniform-structured Hamiltonians in Definition 3.26 can be efficiently implemented in parallel.

Theorem 3.30 (Parallel quantum walks for m-uniform-structured Hamiltonians)**.** *For an m-uniform-structured Hamiltonian, the (r, m)-parallel quantum walk $Q^{(r,m)}$ can be implemented to precision ϵ by a quantum circuit of*

- *depth $O(1)$ and size $O(r)$ w.r.t. queries to $\mathcal{O}_H^b$ with $b = O(\log(m/\epsilon))$,*
- *depth $O(1)$ and size $O(rm)$ w.r.t. queries to $\mathcal{O}_P$, and*
- *depth $O\Big(\log r \cdot \log^2 n + \log^2 \log(m/\epsilon)\Big)$ and size $O\Big(mr^2n^4 + r\log^4(m/\epsilon)\Big)$ w.r.t. gates,*

for $r = \mathrm{polylog}(1/\epsilon)$.

Note that when $m = 1$, the above theorem reduces to Theorem 3.22. For readability, the proof of this theorem is postponed to Appendix A.

4 Parallel LCU for Hamiltonian Series

In this section, we show how to implement a linear combination of block-encoded powers of a Hamiltonian (i.e., a Hamiltonian power series) by a parallel quantum circuit. The method is based on the LCU (linear combination of unitaries) algorithm developed through [3, 10, 16, 30, 32, 36, 116]. Here we adopt the block-encoding version of LCU in [36]. The results in this section will be used to implement a linear combination of parallel quantum walks to approximate the evolution unitary e^{-iHt} in Section 5.

We first recall two technical lemmas from [36]. Although they apply to more general matrices, here for our purpose we restrict them to the Hamiltonians.

Lemma 4.1 (Product of block-encoded Hamiltonian powers [36])**.** *Let K be any Hamiltonian. For $n, m \in \mathbb{N}$, if U is an (α, a, δ)-block-encoding of K^n and V is a (β, b, ϵ)-block-encoding of K^m, then $(\mathbb{1}_a \otimes V)(\mathbb{1}_b \otimes U)$ is an $(\alpha\beta, a + b, \alpha\epsilon + \beta\delta)$-block-encoding of K^{n+m}, where $\mathbb{1}_s$ is the identity operator acting on the proper subsystem composed of the s qubits.*

Definition 4.2 (State preparation unitary)**.** Let $\boldsymbol{a} \in \mathbb{C}^R$ and $\alpha := \|\boldsymbol{a}\|_1$, where $\|\cdot\|_1$ is the l_1 norm. A unitary V is called an (α, ϵ)-state-preparation-unitary of $\boldsymbol{a}$, if the state $V|0\rangle$ is ϵ-close[17] to $\frac{1}{\sqrt{\alpha}}\sum_{r\in[R]}\sqrt{a_r}|r\rangle$ with $|0\rangle \in \mathbb{C}^R$, where a_r is the r^{th} entry of $\boldsymbol{a}$.

This definition is a special case of *state preparation pair* in [36], as shown in Appendix B.

Lemma 4.3 (LCU for Hamiltonian series [36])**.** *Let $\boldsymbol{a} \in \mathbb{C}^R$ be an R-dimensional vector, K be a Hamiltonian, and*

[17] Here, for simplicity of later proofs, the precision is measured by l_2-norm, which is different from the one in [36]. See Appendix B for more details.

- $A := \sum_{r\in[R]} a_r K^r$ *be a power series of* K*;*
- V *be an* (α, δ)*-state-preparation-unitary of* $\boldsymbol{a}$*;*
- U_r *be an* (β, b, ϵ)*-block-encoding of* K^r*.*

If $U := \sum_{r\in[R]} |r\rangle\langle r| \otimes U_r$*, then* $\Big(V^\dagger \otimes \mathbb{1}\Big)U(V \otimes \mathbb{1})$ *is an* $(\alpha\beta, s + b, \alpha\beta R\delta + \alpha\beta\epsilon)$*-block-encoding of* A *with* $s := \lceil \log R \rceil$*.*

Combining Lemma 4.1 and Lemma 4.3, we can obtain the following:

Corollary 4.4. *Let* $\boldsymbol{a} \in \mathbb{C}^R$ *be an* R*-dimensional vector,* K *be a Hamiltonian,* $s := \lceil \log R \rceil$*, and*

- $A := \sum_{r\in[R]} a_r K^r$ *be a power series of* K*;*
- V *be an* (α, δ)*-state-preparation-unitary of* $\boldsymbol{a}$*;*
- W_r *be a* $(1, b_r, \epsilon)$*-block-encoding of* K^r*, and*

$$W := \sum_{r\in[R]} |r\rangle\langle r| \otimes \prod_{j\in[s]} \Big(\mathbb{1}_{b-b_{2^j}} \otimes W_{2^j}\Big)^{r_j} \tag{20}$$

where $b := \sum_{j\in[s]} b_{2^j}$ *and* r_j *is the* j^{th} *bit of* r*.*

Then $\Big(V^\dagger \otimes \mathbb{1}\Big)W(V \otimes \mathbb{1})$ *is an* $(\alpha, s + b, \alpha R\delta + \alpha s\epsilon)$*-block-encoding of* A*.*

Proof. Since W_{2^j} is a $(1, b_{2^j}, \epsilon)$-block-encoding of K^{2^j}, by Lemma 4.1,

$$U_r := \prod_{j\in[s]} \Big(\mathbb{1}_{b-b_{2^j}} \otimes W_{2^j}\Big)^{r_j}$$

is a $(1, b, s\epsilon)$-block-encoding of K^r. Then by Lemma 4.3 we reach the conclusion. □

Now we present two lemmas showing how to implement V and W in Corollary 4.4 by parallel quantum circuits. The idea of Lemma 4.5 is similar to a data structure for matrix sampling in [117], but here we consider complex amplitudes (rather than real amplitudes in [117]) and explicitly compute the gate complexity. This lemma is based on previous works on quantum sampling [93].

Lemma 4.5 (Parallel state preparation)**.** *Let* $\boldsymbol{a} \in \mathbb{C}^R$ *be an* R*-dimensional vector such that for each* $r \in [R]$*,* a_r *is arithmetic-depth-efficiently computable given* r *as input, and let* $\alpha := \|\boldsymbol{a}\|_1$*. An* (α, ϵ)*-state-preparation-unitary* V *of* $\boldsymbol{a}$ *can be implemented by an* $O\Big(\log^3 \log(1/\epsilon)\Big)$*-depth and* $O\Big(\log^5(1/\epsilon)\Big)$*-size quantum circuit for* $R = O(\log(1/\epsilon))$*.*

Proof. Assume $R = 2^s$ w.l.o.g., because one can enlarge the dimension of $\boldsymbol{a}$ by appending enough 0 entries. Let $a_r = e^{i\theta_r}|a_r|$ and partial sum $S(j, l) := \sum_{r=j}^{l-1} |a_r|$. Recall that our goal is to perform the mapping $|0\rangle^{\otimes s} \mapsto \frac{1}{\sqrt{\alpha}} \sum_{r\in[R]} \sqrt{a_r}\,|r\rangle$. The state preparation V consists of s steps, and in the k^{th} step we perform the mapping

$$\begin{cases} |0\rangle^{\otimes s} \mapsto U_{0,0}\,|0\rangle\,|0\rangle^{\otimes s-1} & k = 0, \\ |x\rangle\,|0\rangle^{\otimes s-k} \mapsto |x\rangle\,U_{k,x}\,|0\rangle\,|0\rangle^{\otimes s-k-1} & 1 \le k < s, \end{cases} \tag{21}$$

for all $x \in [2^k]$, where $U_{k,x}$ is a single qubit gate such that

$$U_{k,x}|0\rangle = \sqrt{\gamma_x}|0\rangle + e^{i\beta_x/2}\sqrt{1-\gamma_x}|1\rangle,$$

and we define

$$\gamma_x := \frac{S(u,w)}{S(u,v)}, \qquad \beta_x := \theta_w - \theta_u,$$

with $u := 2^{s-k}x$, $v := 2^{s-k}x + 2^{s-k}$ and $w := (u+v)/2$. A proof of the correctness of this procedure can be found in [117] and [93], except that here we have an additional accumulative phase β_x to handle the complex amplitudes.[18]

Let us compute the gate complexity. To achieve a total precision ϵ, each step of (21) needs to be (ϵ/s)-precise. Each step of (21) consists of two controlled rotations, C_b-R_Y and C_b-R_Z, which are conditioned on elementary arithmetic functions of γ_x and β_x respectively, where $b = O(\log(s/\epsilon)) = O(\log(1/\epsilon))$ is the number of bits for the required precision. While β_x can be easily computed by an arithmetic-depth-efficient quantum circuit, as it is an elementary arithmetic function of a_u and a_w, which are arithmetic-depth-efficiently computable from x; γ_x needs to be computed from the more complicated $S(u,v)$.

Note that one can compute all $S(j,l)$ required at once in an ancilla space. This is done by

- first computing $\left(2^{-b}/R\right)$-precise a_r for all r by arithmetic-depth-efficient circuits on the input $|0,1,\ldots,R-1\rangle$, due to the assumption that a_r is arithmetic-depth-efficiently computable;
- then computing $S(j,l)$ in an inductive way analogous to the proof of Lemma 2.3, except that one needs to create a constant number of copies of each $S(j,l)$ by COPY^{b+s} for future computation.

The input state $|0,1,\ldots,R-1\rangle$ can be prepared by an $O(1)$-depth and $O(Rs)$-size quantum circuit, while the inductive summation procedure requires depth $O\left(s\log^2(b+s)\right)$ and size $O\left(R(b+s)^4\right)$ by Lemma 2.6, Lemma 2.1 and the proof of Lemma 2.3. Thus all $S(j,l)$ required can be computed by an $O\left(\log^3\log(1/\epsilon)\right)$-depth and $O\left(\log^5(1/\epsilon)\right)$-size quantum circuit.

By Lemma 2.2 the controlled rotations in each step of (21) can be implemented by an $O(\log b)$-depth and $O(b)$-size quantum circuit, once the rotation angles, some elementary arithmetic functions of γ_x, β_x are computed by an $O\left(\log^2 b\right)$-depth and $O(b^4)$-size quantum circuit. Summing up the s steps gives the final complexity. □

The next lemma generalizes Lemma 8 in [3] to the block-encoding case.

Lemma 4.6 (Parallel implementation of W)**.** *The unitary W in (20) can be implemented by a quantum circuit of depth $\sum_{j\in[s]} \mathtt{Dep}(W_{2^j})$ and size $\sum_{j\in[s]} \mathtt{Siz}(W_{2^j})$, where* `Dep` *and* `Siz` *refer to the depth and size cost of a subroutine.*

Proof. A parallel quantum circuit implementation of W is shown in Figure 6, where we omit the identity operator $\mathbb{1}_{b-b_{2^j}}$ on the ancilla space for simplicity of notations. Since the controlled version of W_{2^j} has the same complexity as W_{2^j} up to a constant factor, the final complexity then follows easily. □

[18] An alternative way to handle the complex amplitudes is: in the construction of the state preparation unitary, prepare the state $\frac{1}{\sqrt{\alpha}}\sum_{r\in[R]}\sqrt{|a_r|}|r\rangle$; and in the LCU, add all the phases $e^{i\theta_r}$ for $r \in [R]$ in parallel.

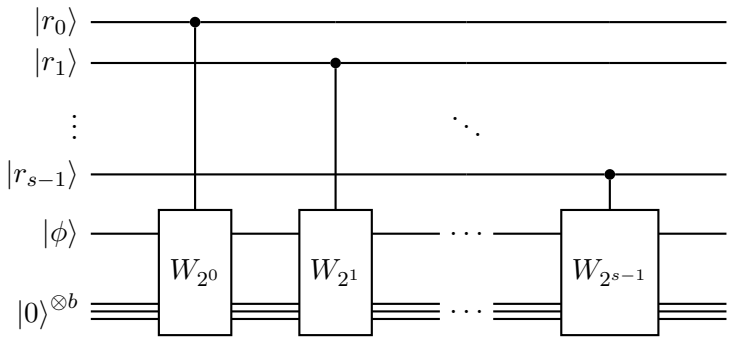

Figure 6: A parallel quantum circuit for W.

5 Parallel Hamiltonian Simulation

Now we are ready to present a parallel quantum simulation algorithm for uniform-structured Hamiltonians by assembling the techniques developed in the previous sections. Following [3, 10, 30], we simulate the evolution unitary e^{-iHt} by first splitting the time interval t into small segments each of length Δt, then approximating the evolution within each segment by the truncated Taylor series

$$e^{-iH\Delta t} \approx \sum_{r=0}^{R-1} \frac{(-iH\Delta t)^r}{r!}. \tag{22}$$

In our setting, Δt should be chosen such that the monomial $(H\Delta t)^r$ can be obtained from the parallel quantum walk in Section 3. Using the results of Section 4 to combine these walk operators, we get a block-encoding of $e^{-iH\Delta t}$, then we apply these $e^{-iH\Delta t}$ sequentially on the initial state. In this way we are able to simulate e^{-iHt}. To guarantee a close to 1 success amplitude after each application of $e^{-iH\Delta t}$, a technique introduced in [10, 30, 31] called robust oblivious amplitude amplification will be used.

Recall that we assume $\|H\|_{\max} = 1$. Since the monomial $\left(\frac{H}{md}\right)^r$ can be obtained from a parallel quantum walk for m-uniform-structured Hamiltonians, we choose $\Delta t := \frac{1}{md}$. By rescaling H with a constant factor $1/2$, the eigenvalue of $H\Delta t$ lies in the interval $[-1/2, 1/2]$. The following lemma shows that to achieve an ϵ-precision in (22) it suffices to choose $R = \lceil \log(1/\epsilon) \rceil$,[19] by taking $z = -iH\Delta t$.

Lemma 5.1. *Assume $R := \lceil \log(1/\epsilon) \rceil \geq 4$ w.l.o.g., we have*

$$\left| e^z - \sum_{r=0}^{R-1} \frac{z^r}{r!} \right| = \left| \sum_{r=R}^{\infty} \frac{z^r}{r!} \right| \leq \epsilon$$

for $z \in \mathbb{C}$ with $|z| \leq 1/2$.

Proof. Since $\sum_{r=R}^{\infty} \frac{z^r}{r!}$ is absolutely convergent, we have $\left|\sum_{r=R}^{\infty} \frac{z^r}{r!}\right| \leq \sum_{r=R}^{\infty} \frac{|z|^r}{r!}$. The RHS of the above equation can be bounded by the Taylor remainder $\frac{e^{\xi}|z|^R}{R!} \leq \frac{1}{2^R} \leq \epsilon$, where $0 \leq \xi \leq |z| \leq 1/2$. □

[19] Actually $R = O\left(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\right)$ is also sufficient (for example, see [30]). Nevertheless, for simplicity, here we choose $R = \lceil \log(1/\epsilon) \rceil$ instead. According to Lemma 5.2, this simplified choice only incurs a constant factor in the depth complexity and a poly-logarithmic factor in the size complexity, which has a negligible impact on our main results.

Let us analyze the complexity of implementing a block-encoding of $e^{-iH\Delta t}$.

Lemma 5.2. *For any m-uniform-structured Hamiltonian H, there exists a unitary $U_{\Delta t}$ that forms an $(\alpha, R(2\lceil\log m\rceil + 4n), \epsilon)$-block-encoding of $e^{-iH\Delta t}$, and can be implemented by a quantum circuit of*

- *depth $O(\log R)$ and size $O(R)$ w.r.t. queries to $\mathcal{O}_H^b$ with $b = O(R + \log m)$,*
- *depth $O(\log R)$ and size $O(mR)$ w.r.t. queries to $\mathcal{O}_P$, and*
- *depth $O\Big(\log^2 R \cdot \log^2 n + \log^3 R\Big)$ and size $O\Big(mR^2n^4 + R(R + \log m)^4\Big)$ w.r.t. gates,*

where $\Delta t := \frac{1}{md}$, $\alpha < e$ is a constant, and $R = \lceil\log(1/\epsilon)\rceil \geq 4$.

Proof. In Corollary 4.4, take $R = \lceil\log(1/\epsilon)\rceil$, $a_r = \frac{(-i)^r}{r!}$, $K = \frac{H}{md}$ and $W_r = Q^{(r,m)}$. Then $\alpha = \sum_{r\in[R]}|a_r| < e$ is a constant, where each a_r is arithmetic-depth-efficiently computable by Lemma 2.6. Assume $R = 2^s$ w.l.o.g. To achieve a total ϵ-precision of $U_{\Delta t}$, in Corollary 4.4, take the precision of V to be $\epsilon/(2\alpha R)$, and the precision of each W_r (i.e., the precision of $Q^{(r,m)}$ in Theorem 3.30) to be $\epsilon/(2\alpha s)$. By Lemma 4.5, the state preparation unitary V can be implemented by an $O\Big(\log^3\log(1/\epsilon)\Big)$-depth and $O\Big(\log^5(1/\epsilon)\Big)$-size quantum circuit. Combining Lemma 4.6 and Theorem 3.30, the implementation of W has

- depth $O(s)$ and size $O(R)$ w.r.t. queries to $\mathcal{O}_H^b$ with $b = O(\log(sm/\epsilon))$,
- depth $O(s)$ and size $O(mR)$ w.r.t. queries to $\mathcal{O}_P$, and
- depth $O\Big(s\Big\lceil\log R \cdot \log^2 n + \log^2\log(sm/\epsilon)\Big\rceil\Big)$ and size $O\Big(mR^2n^4 + R\log^4(sm/\epsilon)\Big)$ w.r.t. gates.

The final complexity follows from summing up these complexities and the assumption $m = \operatorname{poly} n$. Note that the number of ancilla qubits required for the block-encoding is not a tight upper bound. □

To achieve a constant success amplitude after a sequence of $e^{-iH\Delta t}$, we need to amplify the success amplitude to be close to 1 after each application of $e^{-iH\Delta t}$ with robust oblivious amplitude amplification. The following lemma is a special case of Lemma 6 in [10].

Lemma 5.3 (Robust oblivious amplitude amplification)**.** *Given a unitary U that (α, s, ϵ)-block-encodes a unitary V, where $\alpha = \arcsin\Big(\frac{\pi}{2(2l+1)}\Big)$ for some $l \in \mathbb{N}$, one can construct a $(1, s, O(\epsilon))$-block-encoding of V with $O(l)$ uses of U, $U^\dagger$ and $\mathbb{1} - 2\Pi_s$, where $\Pi_s := |0\rangle\langle 0|_s$ is a projector on the ancilla space of s qubits.*

Corollary 5.4. *Given a unitary U that (β, s, ϵ)-block-encodes a unitary V, where $\beta = O(1)$ is a known constant, a $(1, s+1, O(\epsilon))$-block-encoding of V can be implemented by a quantum circuit of*

- *size $O(1)$ w.r.t. U and $U^\dagger$, and*
- *depth $O\Big(\log s + \log^2\log(1/\epsilon)\Big)$ and size $O\Big(s + \log^4(1/\epsilon)\Big)$ w.r.t. gates.*

Proof. By the assumption, $\Big\|V - \beta\Big(\langle 0|^{\otimes s} \otimes \mathbb{1}\Big)U\Big(|0\rangle^{\otimes s} \otimes \mathbb{1}\Big)\Big\| \leq \epsilon$. Pick the minimum l such that $\alpha := \arcsin\Big(\frac{\pi}{2(2l+1)}\Big) \geq \beta$. Let W be a unitary such that

$$W|0\rangle = \frac{1}{\sqrt{\alpha}}\Big(\sqrt{\beta}\,|0\rangle + \sqrt{\alpha^2 - \beta^2}\,|1\rangle\Big),$$

and let $\tilde{U} := \Big(W^\dagger \otimes \mathbb{1}\Big)(\mathbb{1} \otimes U)(W \otimes \mathbb{1})$, then we have

$$\Big\| V - \alpha\Big(\langle 0|^{\otimes s+1} \otimes \mathbb{1}\Big)\tilde{U}\Big(|0\rangle^{\otimes s+1} \otimes \mathbb{1}\Big)\Big\| \leq \epsilon.$$

Note that $\alpha = O(1)$, by Lemma 5.3 we can construct a $(1, s+1, O(\epsilon))$-block-encoding of V with $O(1)$ uses of $\tilde{U}$, $\tilde{U}^\dagger$ and $\mathbb{1} - 2\Pi_{s+1}$. To achieve an $O(\epsilon)$-precision, the implementation of the rotation W has depth $O\Big(\log^2 \log(1/\epsilon)\Big)$ and size $O\Big(\log^4(1/\epsilon)\Big)$ by Lemma 2.2 and Lemma 2.6. The reflection $\mathbb{1} - 2\Pi_{s+1}$ can be implemented by a C_{s+1}-Z gate with an ancilla qubit, which has depth $O(\log s)$ and size $O(s)$ by Corollary 2.5. The final complexity follows from summing up these complexities. □

Now we can give a precise statement of the main result of this paper. An informal version of this theorem was presented as Theorem 1.1.

Theorem 5.5 (Parallel simulation of uniform-structured Hamiltonians)**.** *An m-uniform-structured Hamiltonian $H = \sum_{w\in[m]} H_w$ acting on n qubits with each H_w being d-sparse, can be simulated for time t to precision ϵ ($\epsilon \leq 0.05$) by a quantum circuit of*

- *depth $O(\tau \log \gamma)$ and size $O(\tau\gamma)$ w.r.t. queries to $\mathcal{O}_H^b$ with $b = O(\gamma)$,*
- *depth $O(\tau \log \gamma)$ and size $O(m\tau\gamma)$ w.r.t. queries to $\mathcal{O}_P$, and*
- *depth $O\Big(\tau\Big(\log^2 \gamma \cdot \log^2 n + \log^3 \gamma\Big)\Big)$ and size $O(\tau\gamma^2 \cdot (mn^4 + \gamma^3))$ w.r.t. gates,*

where $\tau := mdt$, $\gamma := \log(\tau/\epsilon)$.

Proof. Let $\Delta t := 1/(md)$. First consider the case when $\tau = t/\Delta t$ is an integer. To achieve a total precision ϵ, applying Lemma 5.2 followed by Corollary 5.4 with precision ϵ/τ gives a $(1, O(\gamma(n + \log m)), O(\epsilon/\tau))$-block-encoding of $e^{-iH\Delta t}$. Repeat the above procedure τ times; that is, using Lemma 4.1 to multiply these block-encoded $e^{-iH\Delta t}$ we obtain a $(1, O(\tau\gamma(n + \log m)), O(\epsilon))$-block-encoding of e^{-iHt}. By properly scaling the precision ϵ we can remove the constant factor in $O(\epsilon)$ and implement e^{-iHt} to precision ϵ with the same overhead up to a constant factor. The final complexity follows from summing up these complexities.

For the case when τ is not an integer, that is, $\tilde{t} := t - [t/\Delta t] \neq 0$, we can independently simulate the last segment for time $\tilde{t}$. This can be done through simulating $\tilde{H} := H\Delta t/\tilde{t}$ instead for time Δt, where the oracle $\mathcal{O}_{\tilde{H}}$ for $\tilde{H}$ is easy to construct from $\mathcal{O}_H$, with at most $O\Big(\log^2 \gamma\Big)$-depth and $O(\gamma^4)$-size of overhead for the required precision by Lemma 2.6. The final complexity is unchanged. □

6 Lower Bounds

In this section, we prove Theorem 1.3 in Section 1.1, which gives a lower bound on the gate depth of simulating a uniform-structured Hamiltonian and implies that the polylog $\log(1/\epsilon)$ factor in the gate depth in Theorem 5.5 cannot be significantly improved to $o(\log \log(1/\epsilon))$. Our proof is based on the proof of Theorem 1.2 in [31], which gives a lower bound that simulating any sparse Hamiltonian to precision ϵ has size $\Omega\Big(\frac{\log(1/\epsilon)}{\log \log(1/\epsilon)}\Big)$ w.r.t. queries, as an extension of the "no-fast-forwarding theorem" in [13]. Their proof basically reduces the problem of computing the parity of N bits (with unbounded error, i.e., with success probability strictly greater than 1/2), to simulating a 2-sparse $2N \times 2N$ Hamiltonian (to a high precision). Our lower bound is achieved by two simple observations: the Hamiltonian

used there is 6-band, which is actually uniform-structured as shown in Lemma 3.12; and computing the parity of N bits with unbounded error requires depth $\Omega(\log N)$.

Proof of Theorem 1.3. We will show that there exists a uniform-structured Hamiltonian H such that simulating H to precision ϵ requires gate depth $\Omega(\log\log(1/\epsilon))$. Following [31], consider a $2N \times 2N$ Hamiltonian H determined by an N-bit string $x = x_0 \dots x_{N-1}$, such that

$$\langle j,k|\, H\, |j-1, k\oplus x_j\rangle = \langle j-1, k\oplus x_j|\, H\, |j,k\rangle = \frac{\sqrt{(N-j+1)j}}{N}$$

for all $j \in [N]$ and $k \in [2]$, where $\oplus$ stands for the XOR operator. Note that here H is 6-band, thus is uniform-structured by Lemma 3.12. Also we have $\|H\|_{\max} \leq 1$, which can be normalized to 1 with a constant overhead. In [31] it is shown that:

1. $\left|\langle N, \mathrm{PARITY}(x)|\, e^{-iHt}\, |0,0\rangle\right| = |\sin(t/N)^N|$, where $\mathrm{PARITY}(x) = x_0 \oplus \dots \oplus x_{N-1}$ is the parity of x;
2. if $N = \Theta\Big(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\Big)$, then there is an unbounded-error algorithm to compute $\mathrm{PARITY}(x)$, by simulating H for a constant time t to precision ϵ on the input $|0,0\rangle$, followed by a computational basis measurement.

Take $N = \Theta\Big(\frac{\log(1/\epsilon)}{\log\log(1/\epsilon)}\Big)$. To finish the proof, it suffices to show that computing $\mathrm{PARITY}(x)$ with unbounded error requires gate depth $\Omega(\log N) = \Omega(\log\log(1/\epsilon))$. This is trivial because $\mathrm{PARITY}(x)$ depends on all x_j for $j \in [N]$, while any $o(\log N)$-depth quantum circuit only involves $o(N)$ input qubits. More precisely, for the sake of contradiction, suppose there exists an $o(\log N)$-depth quantum circuit that takes an input $|x\rangle$ and outputs $\mathrm{PARITY}(x)$ by a measurement on the first output qubit with success probability $> 1/2$, then there must be an input qubit holding $|x_k\rangle$ for some $k \in [N]$ that is not connected to the output qubit by a path of gates and wires. However, $x_k = 0$ and $x_k = 1$ yield different values of $\mathrm{PARITY}(x)$. This gives a contradiction. □

7 Applications

It seems that the parallel quantum algorithm for Hamiltonian simulation developed above can be applied to a wide range of simulation tasks in physics. As a concrete illustration, three examples of physical interest are presented in this section. For each example, we explicitly calculate the total gate complexity of the parallel quantum simulation algorithm for it and compare with the prior art. In particular, we calculate the gate cost for implementing the oracle $\mathcal{O}_H$ and $\mathcal{O}_P$ in the algorithm. Since our choices of $\mathcal{O}_P$ turn out to be efficiently implementable by quantum gates in these examples, compared to the commonly chosen oracle $\mathcal{O}_L$, we believe that our definition of the oracle $\mathcal{O}_P$ is more reasonable in these applications. The results in this section were already summarized in Subsection 1.1 as Table 1.

7.1 Simulation of the Heisenberg Model

Many body localization (MBL) is an intriguing phenomenon in the long-time behaviour of a closed quantum system with disorders and interactions [75, 118, 119]. In contrary to the conventional assumption in quantum statistical mechanics that a system coupling to a bath (i.e., a large environment) after a long time will achieve a thermal equilibrium which erases the initial condition, the MBL system as an isolated many-body quantum system resists

such thermalization — in a local subsystem the information of the initial state is remembered forever. While a theoretical understanding of MBL still remains challenging since its introduction by Anderson in 1958 [120], tremendous numerical works on various systems have been conducted through recent years (for example, see [75, 119, 121] for reviews). A typical example for numeric studies of MBL is one-dimensional Heisenberg model [75–77]. Due to the difficulty of simulating many-body dynamics classically, quantum simulations can investigate properties of MBL in larger systems intractable for classical computers.

Following [77], we consider the problem of simulating the one-dimensional nearest-neighbor Heisenberg model with a random magnetic field in the z direction. More concretely, we will simulate an n-qubit Hamiltonian

$$H = \sum_{w\in[n]} (X_w X_{w+1} + Y_w Y_{w+1} + Z_w Z_{w+1} + h_w Z_w), \tag{23}$$

where $h_w \in [-1, 1]$ is chosen uniformly at random, the subscript w indicates the qubit w that the Pauli matrix acts on, and $w = n$ is equivalent to $w = 0$ by assuming the periodic boundary conditions.

First observe that H is a $(2, n)$-local Hamiltonian with local term $H_w := X_w X_{w+1} + Y_w Y_{w+1} + Z_w Z_{w+1} + h_w Z_w$ for $w \in [n]$. The locality of H_w is indicated by an n-bit string $s(w)$ with the u^{th} bit defined as

$$s(w)_u := [\![w = u \vee w + 1 = u]\!]$$

for all $u \in [n]$, given by the oracle $\mathcal{O}_P$ as in Lemma 3.28. Then by Lemma 3.28, H is m-uniform-structured with $m = n$, thus we can apply Theorem 5.5 to simulate H. For comparison, take the simulation time $t = n$ as in [77]. Since $\|H\|_{\max} \le \sum_{w\in[n]} \|H_w\|_{\max} = O(n)$, to simulate H for time t, it is equivalent to simulate a normalized H with max norm ≤ 1 for a rescaled time $\tilde{t} := \|H\|_{\max} t = O(n^2)$. Thus we can determine the parameters in Theorem 5.5: $\tau = O(n^3)$, $\gamma = O(\log(n/\epsilon))$ and $b = O(\log(n/\epsilon))$.

For the total complexity of the algorithm, one should also compute the gate complexities for implementing the oracle $\mathcal{O}_H^b$ and $\mathcal{O}_P$.

- To implement $\mathcal{O}_P$, that is, to perform the mapping $|w\rangle|0\rangle \mapsto |w\rangle|s(w)\rangle$, one can first prepare the state $|0, \ldots, n-1\rangle$ in the second register, then make n copies of $|w\rangle$ by applying $\mathrm{COPY}_n^{\lceil \log n \rceil}$, and finally calculate each Boolean function $s(w)_u$ for $u \in [n]$ in parallel followed by garbage cleaning. Together with Lemma 2.6, this can be done by an $O(\log n)$-depth and $O(n \log n)$-size quantum circuit.

- To implement $\mathcal{O}_H^b$, one needs to first generate n uniform random $h_w \in [-1, 1]$ to b-bit precision in the pre-processing. This can be done by an $O(1)$-depth and $O(nb)$-size quantum circuit using Hadamard gates followed by single-qubit measurements. The pre-processing only needs to be done once. Later when a h_w is required each time we can apply a COPY^b gate to prepare a new copy of it.

  To perform the mapping $|j, k, 0\rangle \mapsto |j, k, H_{jk}\rangle$, one can calculate an entry H_{jk} by summing up those $(H_w)_{jk}$ with $(j, k) \in \mathsf{H}_w$. Recall that $[\![(j, k) \in \mathsf{H}_w]\!]$ is arithmetic-depth-efficiently computable for uniform-structured H, and $(H_w)_{jk}$ is easy to determine given a copy of h_w, therefore the above procedure can be implemented by an $O\big(\log^2 n + \log b\big)$-depth and $O(n^5 + nb)$-size quantum circuit (in a way analogous to computing $c(j, k)$ in the proof of Lemma A.3).

Thus, we obtain the following:

Corollary 7.1 (Parallel simulation of the Heisenberg model)**.** *The Heisenberg Hamiltonian in* (23) *can be simulated for time $t = n$ to precision ϵ by an $O\Big(n^3 \log^3 n \cdot \log^3 \log(1/\epsilon)\Big)$-depth and $O\Big(n^8 \log^5(n/\epsilon)\Big)$-size quantum circuit.*

In [77] the performance of different quantum simulation algorithms on this task are compared, amongst which the asymptotically best one is based on quantum signal processing [7, 35], which achieves a gate complexity $O\left(n^3 \log n + \frac{\log(1/\epsilon)}{\log\log(1/\epsilon)} \cdot n \log n\right)$. Later a quantum algorithm to simulate lattice Hamiltonians [37] shows a better dependence on n in the gate complexity, with depth $O(n \operatorname{polylog}(n/\epsilon))$ and size $O(n^2 \operatorname{polylog}(n/\epsilon))$. We note that in the gate depth all these works have a $\operatorname{polylog}(1/\epsilon)$ dependence, while Corollary 7.1 only contains a $\operatorname{polylog}\log(1/\epsilon)$ factor.

7.2 Simulation of the Sachdev-Ye-Kitaev Model

The Sachdev-Ye-Kitaev (SYK) model [78, 79], a simple but important exactly solvable many-body system, has drawn an increasing interest in the condense matter physics and high energy physics communities due to its many striking properties [78–80, 122, 123] and its potential to have an interesting holographic dual [79, 80]. Like the MBL problem in Section 7.1, numeric studies of a larger SYK model enabled by quantum simulation could extend our understandings about its features and dual interpretation.

Following [81,85], we consider the problem of simulating the SYK model evolving under a Hamiltonian

$$H = \frac{1}{4 \cdot 4!} \sum_{p,q,r,s=0}^{2n-1} J_{pqrs} \gamma_p \gamma_q \gamma_r \gamma_s, \tag{24}$$

where each $J_{pqrs} \sim \mathcal{N}(0, \sigma^2)$ is chosen randomly from a normal distribution with variance $\sigma^2 = \frac{3! J^2}{(2n)^3}$ (J is assumed to be a constant), and γ_p are Majorana fermion operators such that $\{\gamma_p, \gamma_q\} = 2[\![p = q]\!]\mathbb{1}$. The Majorana operator can be expressed as a tensor product of Pauli matrices by the Jordan-Wigner transformation

$$\gamma_p \mapsto \begin{cases} Z_0 \dots Z_{p/2-1} X_{p/2}, & p \text{ is even}, \\ Z_0 \dots Z_{(p-3)/2} Y_{(p-1)/2}, & p \text{ is odd} \end{cases} \tag{25}$$

for $p \in [2n]$ (unlike in [81], our index starts from 0), where as usual the subscript of a Pauli matrix indicates the qubit it acts on. Now H can be expressed as a Pauli sum on n qubits:

$$H = \alpha \sum_{w \in [2n]^4} J_w H_w \tag{26}$$

where α is a constant, J_w is chosen randomly from a normal distribution, and each H_w is a tensor product of Pauli matrices.

It can be seen by Lemma 3.27 that the SYK Hamiltonian in (26) is m-uniform-structured with $m = (2n)^4$, thus we can apply Theorem 5.5 to simulate it. Note that

$$\|H\|_{\max} \le \sum_{w \in [2n]^4} \alpha \|J_w H_w\|_{\max} \le \sum_{w \in [2n]^4} \alpha \, \mathbf{E}[|J_w|] = \alpha (2n)^4 \cdot \frac{\sqrt{2}\sigma}{\sqrt{\pi}} = O\Big(n^{2.5}\Big)$$

where we use $J_w \sim \mathcal{N}(0, \sigma^2)$ with $\sigma^2 = O(1/n^3)$. Thus, to simulate H for time t, it is equivalent to simulate its normalized Hamiltonian with max norm ≤ 1 for a rescaled time $\tilde{t} := \|H\|_{\max} t = O(n^{2.5} t)$. We can determine the parameters in Theorem 5.5: $\tau = O(n^{6.5} t)$, $\gamma = O(\log(nt/\epsilon))$ and $b = O(\log(nt/\epsilon))$.

For the total complexity of the algorithm, we starts from defining a function $\mathcal{P} : [2n]^4 \to \{1, i, -1, -i\} \times \{\mathbb{1}, X, Y, Z\}^n \simeq [4]^{n+1}$ that maps $w \in [m]$ to the Pauli string (with a global phase) of H_w. For example, if $H_w = -iX \otimes Z \otimes Z$ then $\mathcal{P}(w) = (-i, X, Z, Z)$. One can also write the Jordan-Wigner transformation in (25) as a function $\mathcal{J} : [2n] \to \{\mathbb{1}, X, Y, Z\}^n \simeq [4]^n$, which can be computed by an $O(\log n)$-depth and $O(n \log n)$-size quantum circuit. To see this, that is, to perform the mapping $|p\rangle |0\rangle \mapsto |p\rangle |\mathcal{J}(p)\rangle$, we can first prepare the state $|0, \ldots, n-1\rangle$ in the second register, then make n copies of $|p\rangle$, and finally compute the q^{th} bit of $\mathcal{J}(p)$:

$$\mathcal{J}(p)_q := \begin{cases} 3[\![q < p/2]\!] + [\![q = p/2]\!], & p \text{ is even}, \\ 3[\![q < (p-1)/2]\!] + 2[\![q = (p-1)/2]\!], & p \text{ is odd} \end{cases} \in [4]$$

for all $q \in [n]$ in parallel followed by garbage cleaning. By Lemma 2.6, this can be done by an $O(\log n)$-depth and $O(n \log n)$-size quantum circuit. Since for all $(p, q, r, s) \in [2n]^4$, the function $\mathcal{P}(p, q, r, s)$ can be computed from $\mathcal{J}(p), \mathcal{J}(q), \mathcal{J}(r), \mathcal{J}(s)$ by calculating Pauli algebra on each entry of these n-tuples and gathering up the global phases, $\mathcal{P}(w)$ can be computed by an $O(\log n)$-depth and $O(n \log n)$-size quantum circuit.

Now we will determine the gate complexities for implementing the oracle $\mathcal{O}_H^b$ and $\mathcal{O}_P$:

- To implement $\mathcal{O}_P$, note that $s(w)$ in Lemma 3.27 is easily computable from $\mathcal{P}(w)$ by function restriction, thus $\mathcal{O}_P$ can be implemented by an $O(\log n)$-depth and $O(n \log n)$-size quantum circuit.

- To implement $\mathcal{O}_H^b$, one needs to first generate m independent $J_w \in \mathcal{N}(0, \sigma^2)$ to b-bit precision in the pre-processing. This can be done by the Box-Muller transform [124], which derives two independent standard normally distributed variables from two independent uniformly distributed variables in $[0, 1]$ by elementary arithmetic. Thus it suffices to generate m uniformly random variables in $[0, 1]$ to b-bit precision by an $O(1)$-depth and $O(n^4 b)$-size quantum circuit using Hadamard gates followed by single-qubit measurements, and then use Lemma 2.6 to perform the Box-Muller transform with a scaling σ by an $O\left(\log^2 b\right)$-depth and $O(n^4 b^4)$-size quantum circuit. The pre-processing only needs to be done once. Later when a J_w is required each time we can apply a COPY^b gate to make a new copy of it.

  We perform the mapping $|j, k, 0\rangle \mapsto |j, k, H_{jk}\rangle$ in a way similar to the one in Section 7.1; that is, sum up those $(H_w)_{jk}$ with $(j, k) \in \mathsf{H}_w$. Recall that each $[\![(j, k) \in \mathsf{H}_w]\!]$ is arithmetic-depth-efficiently computable for uniform-structured H, and $(H_w)_{jk}$ is easy to compute from the function $\mathcal{P}(w)$ and a copy of J_w. Therefore, the above procedure can be implemented by an $O\left(\log^2 n + \log b\right)$-depth and $O(n^8 + n^4 b)$-size quantum circuit (in a way analogous to the proof of Lemma A.3).

Applying Theorem 5.5, we obtain the following:

Corollary 7.2 (Parallel simulation of the SYK model)**.** *The SYK model defined in* (24) *can be simulated for time t to precision ϵ by an $O\left(n^{6.5} \log^3 n \cdot t \log^3 \log(t/\epsilon)\right)$-depth and $O\left(n^{14.5} t \log^5(nt/\epsilon)\right)$-size quantum circuit.*

The prior best algorithm based on asymmetric qubitization [85] for this task has gate complexity $O(n^{3.5} t + n^{2.5} t \operatorname{polylog}(n/\epsilon))$, which improves the product-formula-based algorithm [81] with gate complexity $O(n^{10} t^2/\epsilon)$. Later work [125] proposes a sparse SYK model also of physical interest and gives a simulation algorithm based on qubitization

with a polylog$(1/\epsilon)$ dependence in the gate complexity. Compared to these works, by introducing parallelism Corollary 7.2 only has a polylog $\log(1/\epsilon)$ dependence in the gate depth.

7.3 Simulation of Quantum Chemistry in Second Quantization

One of the most attractive prospects of quantum simulation is in quantum chemistry to study the static and dynamic properties of chemicals [82, 83, 126]. Prior works on quantum simulation algorithm for chemistry mainly focus on exploiting the special structure of molecular Hamiltonians (e.g. based on first or second quantization, using Gaussian orbital bases or plane wave bases, etc.) to obtain better complexities (see [83] for a review). Compared to these fault-tolerant quantum algorithms, diverse variational quantum algorithms to circumvent a direct simulation (e.g. variational quantum eigensolver [127]) also have been explored in recent years due to their potential for immediate applications in the NISQ era [128].

Following [84] we consider the problem of simulating a molecular electronic structure Hamiltonian in the second-quantized form. In this form, we will simulate a Hamiltonian

$$H = \sum_{p,q\in[n]} h_{pq} a_p^\dagger a_q + \frac{1}{2} \sum_{p,q,r,s\in[n]} h_{pqrs} a_p^\dagger a_q^\dagger a_r a_s \tag{27}$$

where n represents the number of spin orbitals, h_{pq}, h_{pqrs} are one-electron and two-electron integrals, and $a_p^\dagger, a_p$ are fermionic creation and annihilation operators satisfying the relations

$$\left\{a_p^\dagger, a_q\right\} = [\![p = q]\!]\mathbb{1}, \qquad \left\{a_p^\dagger, a_q^\dagger\right\} = \{a_p, a_q\} = 0.$$

As in the "database" algorithm [84], we assume that these b-bit precise h_{pq} and h_{pqrs} are precomputed and stored in a database; for example, an $O((nb)^4)$-size quantum-read/classical-write RAM (QCRAM), to which one quantum access can be done by an $O(\log(nb))$-depth and $O((nb)^4)$-size quantum circuit.

One can apply Jordan-Wigner transformation to the creation and annihilation operators $a_p^\dagger, a_p$ to obtain Hamiltonians acting on qubits:

$$a_p^\dagger \mapsto \frac{1}{2} Z_0 \dots Z_{p-2}(X_{p-1} - iY_{p-1}) \tag{28}$$

$$a_p \mapsto \frac{1}{2} Z_0 \dots Z_{p-2}(X_{p-1} + iY_{p-1}) \tag{29}$$

for $p \in [n]$, where as usual the subscript of a Pauli matrix indicates the qubit it acts on. Note that each resulting qubit Hamiltonian in (28) and (29) is a sum of two tensor products of Pauli matrices, hence by splitting them and applying the transformation to (27) we obtain H as a Pauli sum on n qubits:

$$H = \sum_{w\in[m]} h_w H_w, \tag{30}$$

where $m = O(n^4)$, each h_w is some h_{pq} or h_{pqrs} in (27) (up to a constant factor), and each H_w is a tensor product of Pauli matrices. Here we omit the explicit mapping from the indices p, q, r, s in (27) to the index w in (30), but mention that the mapping can be efficiently performed by an $O(1)$-depth quantum circuit.

Similar to the SYK model, we see by Lemma 3.27 that the molecular Hamiltonian H in (30) is m-uniform-structured with $m = O(n^4)$. According to [84], we have $\sum_{w\in[m]} |h_w| =$

$O(n^4)$. Thus the max norm of H is bounded by

$$\|H\|_{\max} \leq \sum_{w\in[m]} \|h_w H_w\|_{\max} \leq \sum_{w\in[m]} |h_w| = O\Big(n^4\Big).$$

To simulate H for time t it is equivalent to simulate its normalized Hamiltonian for a rescaled time $\tilde{t} := \|H\|_{\max} t = O(n^4 t)$. The parameters in Theorem 5.5 can be determined as follows: $\tau = O(n^8 t)$, $\gamma = O(\log(nt/\epsilon))$ and $b = O(\log(nt/\epsilon))$.

Now let us compute the gate complexities for implementing the oracle $\mathcal{O}_H^b$ and $\mathcal{O}_P$:

- The construction of $\mathcal{O}_P$ is similar to the one in Section 7.2. Here we will omit the details and claim that it can be implemented by an $O(\log n)$-depth and $O(n \log n)$-size quantum circuit.
- To implement $\mathcal{O}_H^b$, that is, to perform the mapping $|j,k,0\rangle \mapsto |j,k,H_{jk}\rangle$, we sum up those $(H_w)_{jk}$ with $(j,k) \in \mathsf{H}_w$ to obtain H_{jk}. Recall that $[\![(j,k) \in \mathsf{H}_w]\!]$ is arithmetic-depth-efficiently computable for uniform-structured H. Also $(H_w)_{jk}$ is easy to compute given: a copy of h_w which can be read out from the database by an $O(\log(nb))$-depth and $O((nb)^4)$-size quantum circuit; together with a string characterizing the Pauli string of H_w (like the function $\mathcal{P}(w)$ for the SYK model) computable by an $O(\log n)$-depth and $O(n \log n)$-size quantum circuit. Therefore the above procedure can be implemented by an $O\Big(\log^2 n + \log b\Big)$-depth and $O(n^8 b^4)$-size quantum circuit (in a way analogous to the proof of Lemma A.3).

As an application of Theorem 5.5, we have the following:

Corollary 7.3 (Parallel simulation of molecular Hamiltonians)**.** *The molecular Hamiltonian in (27) can be simulated for time t to precision ϵ by an $O\Big(n^8 \log^3 n \cdot t \log^3 \log(t/\epsilon)\Big)$-depth and $O\Big(n^{16} t \log^5(nt/\epsilon)\Big)$-size quantum circuit.*

Prior work [84] gives a quantum simulation algorithm for molecular Hamiltonians with gate complexity $O\Big(\frac{n^8 t \log(nt/\epsilon)}{\log\log(nt/\epsilon)}\Big)$. Later algorithmic improvements (e.g. [86–90]) focus on parameters other than the precision ϵ, and all these works have a poly-logarithmic dependence on ϵ as the Hamiltonian simulation subroutines used there have such dependence. By allowing parallelism, Corollary 7.3 exponentially improves the dependence to $\mathrm{polylog}\log(1/\epsilon)$ with respect to the depth.

Acknowledgements

We thank Zhengfeng Ji for pointing out that local Hamiltonians can be represented as Pauli sums. We acknowledge an anonymous reviewer for raising the open question (in Section 1.4) about whether our method can be generalized for simulating time-dependent Hamiltonians via the Dyson series. Zhicheng Zhang also thanks Rolando D. Somma for mentioning that the results in [94] do not concern the depth complexity.

This work was supported in part by the National Natural Science Foundation of China under Grant 61832015. Zhicheng Zhang was supported by the Sydney Quantum Academy, NSW, Australia. Qisheng Wang was also supported in part by the MEXT Quantum Leap Flagship Program (MEXT Q-LEAP) grants No. JPMXS0120319794.

References

[1] Richard P. Feynman. "Simulating physics with computers". International Journal of Theoretical Physics **21**, 467–488 (1982).

[2] Seth Lloyd. "Universal quantum simulators". Science **273**, 1073–1078 (1996).

[3] Andrew M. Childs, Robin Kothari, and Rolando D. Somma. "Quantum algorithm for systems of linear equations with exponentially improved dependence on precision". SIAM Journal on Computing **46**, 1920–1950 (2017).

[4] Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. "Quantum SDP-solvers: Better upper and lower bounds". Quantum **4**, 230 (2020).

[5] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. "A quantum approximate optimization algorithm" (2014). arXiv:1411.4028.

[6] Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. "The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation". In Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP '19). Volume 132, pages 33:1–33:14. (2019).

[7] Guang Hao Low and Isaac L. Chuang. "Hamiltonian simulation by qubitization". Quantum **3**, 163 (2019).

[8] Andrew M. Childs. "On the relationship between continuous- and discrete-time quantum walk". Communications in Mathematical Physics **294**, 581–603 (2009).

[9] Dominic W. Berry and Andrew M. Childs. "Black-box Hamiltonian simulation and unitary implementation". Quantum Information & Computation **12**, 29–62 (2012).

[10] Dominic W. Berry, Andrew M. Childs, and Robin Kothari. "Hamiltonian simulation with nearly optimal dependence on all parameters". In Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS '15). Pages 792–809. (2015).

[11] Lucas Lamata, Adrian Parra-Rodriguez, Mikel Sanz, and Enrique Solano. "Digital-analog quantum simulations with superconducting circuits". Advances in Physics: X **3**, 1457981 (2018).

[12] Dorit Aharonov and Amnon Ta-Shma. "Adiabatic quantum state generation". SIAM Journal on Computing **37**, 47–82 (2007).

[13] Dominic W. Berry, Graeme Ahokas, Richard Cleve, and Barry C. Sanders. "Efficient quantum algorithms for simulating sparse Hamiltonians". Communications in Mathematical Physics **270**, 359–371 (2006).

[14] Nathan Wiebe, Dominic W. Berry, Peter Høyer, and Barry C. Sanders. "Higher order decompositions of ordered operator exponentials". Journal of Physics A: Mathematical and Theoretical **43**, 065203 (2010).

[15] Andrew M. Childs and Robin Kothari. "Simulating sparse Hamiltonians with star decompositions". In Theory of Quantum Computation, Communication, and Cryptography (TQC '10). Pages 94–103. Springer Berlin Heidelberg (2011).

[16] Andrew M. Childs and Nathan Wiebe. "Hamiltonian simulation using linear combinations of unitary operations". Quantum Information & Computation **12**, 901–924 (2012).

[17] Guang Hao Low, Vadym Kliuchnikov, and Nathan Wiebe. "Well-conditioned multiproduct Hamiltonian simulation" (2019). arXiv:1907.11679.

[18] Andrew M. Childs and Yuan Su. "Nearly optimal lattice simulation by product formulas". Physical Review Letters **123**, 050503 (2019).

[19] Earl Campbell. "Random compiler for fast Hamiltonian simulation". Physical Review Letters **123**, 070503 (2019).

[20] Andrew M. Childs, Aaron Ostrander, and Yuan Su. "Faster quantum simulation by randomization". Quantum **3**, 182 (2019).

[21] Yingkai Ouyang, David R. White, and Earl T. Campbell. "Compilation by stochastic Hamiltonian sparsification". Quantum **4**, 235 (2020).

[22] Chi-Fang Chen, Hsin-Yuan Huang, Richard Kueng, and Joel A. Tropp. "Concentration for random product formulas". PRX Quantum **2**, 040305 (2021).

[23] Yuan Su, Hsin-Yuan Huang, and Earl T. Campbell. "Nearly tight Trotterization of interacting electrons". Quantum **5**, 495 (2021).

[24] Paul K. Faehrmann, Mark Steudtner, Richard Kueng, Mária Kieferová, and Jens Eisert. "Randomizing multi-product formulas for improved Hamiltonian simulation". Quantum **6**, 806 (2022).

[25] Matthew Hagan and Nathan Wiebe. "Composite quantum simulations". Quantum **7**, 1181 (2023).

[26] Chien Hung Cho, Dominic W. Berry, and Min-Hsiu Hsieh. "Doubling the order of approximation via the randomized product formula" (2022). arXiv:2210.11281.

[27] Guang Hao Low, Yuan Su, Yu Tong, and Minh C. Tran. "Complexity of implementing Trotter steps". PRX Quantum **4**, 020323 (2023).

[28] Pei Zeng, Jinzhao Sun, Liang Jiang, and Qi Zhao. "Simple and high-precision Hamiltonian simulation by compensating Trotter error with linear combination of unitary operations" (2022). arXiv:2212.04566.

[29] Gumaro Rendon, Jacob Watkins, and Nathan Wiebe. "Improved error scaling for Trotter simulations through extrapolation" (2022). arXiv:2212.14144.

[30] Dominic W. Berry, Andrew M. Childs, Richard Cleve, Robin Kothari, and Rolando D. Somma. "Simulating Hamiltonian dynamics with a truncated Taylor series". Physical Review Letters **114**, 090502 (2015).

[31] Dominic W. Berry, Andrew M. Childs, Richard Cleve, Robin Kothari, and Rolando D. Somma. "Exponential improvement in precision for simulating sparse Hamiltonians". In Proceedings of the 46th Annual ACM SIGACT Symposium on Theory of Computing (STOC '14). Pages 283–292. (2014).

[32] Robin Kothari. "Efficient algorithms in quantum query complexity". PhD thesis. University of Waterloo. (2014). url: http://hdl.handle.net/10012/8625.

[33] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. "Quantum algorithm for linear systems of equations". Physical Review Letters **103**, 150502 (2009).

[34] Guang Hao Low, Theodore J. Yoder, and Isaac L. Chuang. "Methodology of resonant equiangular composite quantum gates". Physical Review X **6**, 041067 (2016).

[35] Guang Hao Low and Isaac L. Chuang. "Optimal Hamiltonian simulation by quantum signal processing". Physical Review Letters **118**, 010501 (2017).

[36] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. "Quantum singular value transformation and beyond: Exponential improvements for quantum matrix arithmetics". In Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (STOC '19). Page 193–204. (2019).

[37] Jeongwan Haah, Matthew B. Hastings, Robin Kothari, and Guang Hao Low. "Quantum algorithm for simulating real time evolution of lattice Hamiltonians". SIAM Journal on Computing **0**, FOCS18–250–FOCS18–284 (2018).

[38] Guang Hao Low and Nathan Wiebe. "Hamiltonian simulation in the interaction picture" (2019). arXiv:1805.00675.

[39] Guang Hao Low. "Hamiltonian simulation with nearly optimal dependence on spectral norm". In Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (STOC '19). Pages 491–502. (2019).

[40] John M. Martyn, Yuan Liu, Zachary E. Chin, and Isaac L. Chuang. "Efficient fully-coherent quantum signal processing algorithms for real-time dynamics simulation". The Journal of Chemical Physics **158**, 024106 (2023).

[41] Qi Zhao, You Zhou, Alexander F. Shaw, Tongyang Li, and Andrew M. Childs. "Hamiltonian simulation with random inputs". Physical Review Letters **129**, 270502 (2022).

[42] Richard Cleve and John Watrous. "Fast parallel circuits for the quantum Fourier transform". In Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science (FOCS '00). Pages 526–536. (2000).

[43] Peter W. Shor. "Algorithms for quantum computation: discrete logarithms and factoring". In Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science (FOCS '94). Pages 124–134. (1994).

[44] Paul Pham and Krysta M. Svore. "A 2D nearest-neighbor quantum architecture for factoring in polylogarithmic depth". Quantum Information & Computation **13**, 937–962 (2013).

[45] Martin Rötteler and Rainer Steinwandt. "A quantum circuit to find discrete logarithms on ordinary binary elliptic curves in depth $O(\log^2 n)$". Quantum Information & Computation **14**, 888–900 (2014).

[46] Lov K. Grover. "A fast quantum mechanical algorithm for database search". In Proceedings of the 28th Annual ACM SIGACT Symposium on Theory of Computing (STOC '96). Pages 212–219. (1996).

[47] Christof Zalka. "Grover's quantum searching algorithm is optimal". Physical Review A **60**, 2746–2751 (1999).

[48] Robert M. Gingrich, Colin P. Williams, and Nicolas J. Cerf. "Generalized quantum search with parallelism". Physical Review A **61**, 052313 (2000).

[49] Lov K. Grover and Jaikumar Radhakrishnan. "Quantum search for multiple items using parallel queries" (2004). arXiv:quant-ph/0407217.

[50] Stacey Jeffery, Frédéric Magniez, and Ronald de Wolf. "Optimal parallel quantum query algorithms". Algorithmica **79**, 509–529 (2017).

[51] Paul Burchard. "Lower bounds for parallel quantum counting" (2019). arXiv:1910.04555.

[52] Tudor Giurgica-Tiron, Iordanis Kerenidis, Farrokh Labib, Anupam Prakash, and William Zeng. "Low depth algorithms for quantum amplitude estimation". Quantum **6**, 745 (2022).

[53] Frederic Green, Steven Homer, and Christopher Pollett. "On the complexity of quantum ACC". In Proceedings 15th Annual IEEE Conference on Computational Complexity (CCC '00). Pages 250–262. (2000).

[54] Cristopher Moore and Martin Nilsson. "Parallel quantum computation and quantum codes". SIAM Journal on Computing **31**, 799–815 (2002).

[55] Frederic Green, Steven Homer, Cristopher Moore, and Christopher Pollett. "Counting, fanout and the complexity of quantum ACC". Quantum Information & Computation **2**, 35–65 (2002).

[56] Barbara M. Terhal and David P. DiVincenzo. "Adaptive quantum computation, constant depth quantum circuits and Arthur-Merlin games". Quantum Information & Computation **4**, 134–145 (2004).

[57] Stephen Fenner, Frederic Green, Steven Homer, and Yong Zhang. "Bounds on the power of constant-depth quantum circuits". In Proceedings of the 15th International Conference on Fundamentals of Computation Theory (FCT '05). Pages 44–55. (2005).

[58] Peter Høyer and Robert Špalek. "Quantum fan-out is powerful". Theory of Computing **1**, 81–103 (2005).

[59] Debajyoti Bera, Frederic Green, and Steven Homer. "Small depth quantum circuits". SIGACT News **38**, 35–50 (2007).

[60] Yasuhiro Takahashi and Seiichiro Tani. "Collapse of the hierarchy of constant-depth exact quantum circuits". Computational Complexity **25**, 849–881 (2016).

[61] Matthew Coudron and Sanketh Menda. "Computations with greater quantum depth are strictly more powerful (relative to an oracle)". In Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing (STOC '20). Pages 889–901. (2020).

[62] Nai-Hui Chia, Kai-Min Chung, and Ching-Yi Lai. "On the need for large quantum depth". Journal of the ACM**70** (2023).

[63] Jiaqing Jiang, Xiaoming Sun, Shang-Hua Teng, Bujiao Wu, Kewen Wu, and Jialin Zhang. "Optimal space-depth trade-off of CNOT circuits in quantum logic synthesis". In Proceedings of the 31st Annual ACM SIAM Symposium on Discrete Algorithms (SODA '20). Pages 213–229. (2020).

[64] Sergey Bravyi, David Gosset, and Robert König. "Quantum advantage with shallow circuits". Science **362**, 308–311 (2018).

[65] Adam Bene Watts, Robin Kothari, Luke Schaeffer, and Avishay Tal. "Exponential separation between shallow quantum circuits and unbounded fan-in shallow classical circuits". In Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (STOC '19). Pages 515–526. (2019).

[66] François Le Gall. "Average-case quantum advantage with shallow circuits". In Proceedings of the 34th Computational Complexity Conference (CCC '19). Pages 1–20. (2019).

[67] Sergey Bravyi, David Gosset, Robert König, and Marco Tomamichel. "Quantum advantage with noisy shallow circuits". Nature Physics **16**, 1040–1045 (2020).

[68] Yihui Quek, Mark M. Wilde, and Eneet Kaur. "Multivariate trace estimation in constant quantum depth" (2022). arXiv:2206.15405.

[69] Richard Jozsa. "An introduction to measurement based quantum computation" (2005). arXiv:quant-ph/0508124.

[70] Anne Broadbent and Elham Kashefi. "Parallelizing quantum circuits". Theoretical Computer Science **410**, 2489–2510 (2009).

[71] Dan Browne, Elham Kashefi, and Simon Perdrix. "Computational depth complexity of measurement-based quantum computation". In Theory of Quantum Computation, Communication, and Cryptography (TQC '10). Volume 6519, pages 35–46. (2011).

[72] Robert Beals, Stephen Brierley, Oliver Gray, Aram W. Harrow, Samuel Kutin, Noah Linden, Dan Shepherd, and Mark Stather. "Efficient distributed quantum computing". Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **469**, 20120686 (2013).

[73] Mingsheng Ying and Yuan Feng. "An algebraic language for distributed quantum computing". IEEE Transactions on Computers **58**, 728–743 (2009).

[74] Mingsheng Ying, Li Zhou, and Yangjia Li. "Reasoning about parallel quantum programs" (2019). arXiv:1810.11334.

[75] Rahul Nandkishore and David A. Huse. "Many-body localization and thermalization in quantum statistical mechanics". Annual Review of Condensed Matter Physics **6**, 15–38 (2015).

[76] David J. Luitz, Nicolas Laflorencie, and Fabien Alet. "Many-body localization edge in the random-field Heisenberg chain". Physical Review B **91**, 081103 (2015).

[77] Andrew M. Childs, Dmitri Maslov, Yunseong Nam, Neil J. Ross, and Yuan Su. "Toward the first quantum simulation with quantum speedup". Proceedings of the National Academy of Sciences **115**, 9456–9461 (2018).

[78] Subir Sachdev and Jinwu Ye. "Gapless spin-fluid ground state in a random quantum Heisenberg magnet". Physical Review Letters **70**, 3339–3342 (1993).

[79] Alexei Y. Kitaev. "A simple model of quantum holography". Talks at KITP, April 7, 2015 and May 27, 2015.

[80] Juan Maldacena and Douglas Stanford. "Remarks on the Sachdev-Ye-Kitaev model". Physical Review D **94**, 106002 (2016).

[81] Laura García-Álvarez, Íñigo Luis Egusquiza, Lucas Lamata, Adolfo del Campo, Julian Sonner, and Enrique Solano. "Digital quantum simulation of minimal AdS/CFT". Physical Review Letters **119**, 040501 (2017).

[82] Man-Hong Yung, James D. Whitfield, Sergio Boixo, David G. Tempel, and Alán Aspuru-Guzik. "Introduction to quantum algorithms for physics and chemistry". In Advances in Chemical Physics. Pages 67–106. John Wiley & Sons, Inc. (2014).

[83] Bela Bauer, Sergey Bravyi, Mario Motta, and Garnet Kin-Lic Chan. "Quantum algorithms for quantum chemistry and quantum materials science". Chemical Reviews **120**, 12685–12717 (2020).

[84] Ryan Babbush, Dominic W. Berry, Ian D. Kivlichan, Annie Y. Wei, Peter J. Love, and Alán Aspuru-Guzik. "Exponentially more precise quantum simulation of fermions in second quantization". New Journal of Physics **18**, 033032 (2016).

[85] Ryan Babbush, Dominic W. Berry, and Hartmut Neven. "Quantum simulation of the Sachdev-Ye-Kitaev model by asymmetric qubitization". Physical Review A **99**, 040301 (2019).

[86] Ryan Babbush, Dominic W. Berry, Yuval R. Sanders, Ian D. Kivlichan, Artur Scherer, Annie Y. Wei, Peter J. Love, and Alán Aspuru-Guzik. "Exponentially more precise quantum simulation of fermions in the configuration interaction representation". Quantum Science and Technology **3**, 015006 (2017).

[87] Ryan Babbush, Nathan Wiebe, Jarrod McClean, James McClain, Hartmut Neven, and Garnet Kin-Lic Chan. "Low-depth quantum simulation of materials". Physic Review X **8**, 011044 (2018).

[88] Ian D. Kivlichan, Jarrod McClean, Nathan Wiebe, Craig Gidney, Alán Aspuru-Guzik, Garnet Kin-Lic Chan, and Ryan Babbush. "Quantum simulation of electronic structure with linear depth and connectivity". Physical Review Letters **120**, 110501 (2018).

[89] Ryan Babbush, Dominic W. Berry, Jarrod R. McClean, and Hartmut Neven. "Quantum simulation of chemistry with sublinear scaling in basis size". npj Quantum Information**5** (2019).

[90] Dominic W. Berry, Craig Gidney, Mario Motta, Jarrod R. McClean, and Ryan Babbush. "Qubitization of arbitrary basis quantum chemistry leveraging sparsity and low rank factorization". Quantum **3**, 208 (2019).

[91] Charles H. Bennett. "Logical reversibility of computation". IBM Journal of Research and Development **17**, 525–532 (1973).

[92] Michael A. Nielsen and Isaac L. Chuang. "Quantum computation and quantum information: 10th anniversary edition". Cambridge University Press. (2010).

[93] Lov K. Grover and Terry Rudolph. "Creating superpositions that correspond to efficiently integrable probability distributions" (2002). arXiv:quant-ph/0208112.

[94] Yosi Atia and Dorit Aharonov. "Fast-forwarding of Hamiltonians and exponentially precise measurements". Nature Communications**8** (2017).

[95] Shouzhen Gu, Rolando D. Somma, and Burak Şahinoğlu. "Fast-forwarding quantum evolution". Quantum **5**, 577 (2021).

[96] Frédéric Magniez, Ashwin Nayak, Jérémie Roland, and Miklos Santha. "Search via quantum walk". SIAM Journal on Computing **40**, 142–164 (2011).

[97] Xiao-Ming Zhang, Tongyang Li, and Xiao Yuan. "Quantum state preparation with optimal circuit depth: implementations and applications". Physical Review Letters **129**, 230504 (2022).

[98] Xiaoming Sun, Guojing Tian, Shuai Yang, Pei Yuan, and Shengyu Zhang. "Asymptotically optimal circuit depth for quantum state preparation and general unitary synthesis". IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **42**, 3301–3314 (2023).

[99] Gregory Rosenthal. "Query and depth upper bounds for quantum unitaries via Grover search" (2021). arXiv:2111.07992.

[100] Pei Yuan and Shengyu Zhang. "Optimal (controlled) quantum state preparation and improved unitary synthesis by quantum circuits with any number of ancillary qubits". Quantum **7**, 956 (2023).

[101] Nai-Hui Chia, Kai-Min Chung, Yao-Ching Hsieh, Han-Hsuan Lin, Yao-Ting Lin, and Yu-Ching Shen. "On the impossibility of general parallel fast-forwarding of Hamiltonian simulation". In Proceedings of the Conference on Proceedings of the 38th Computational Complexity Conference (CCC '23). Pages 1–45. (2023).

[102] Mihir Bellare and Phillip Rogaway. "Random oracles are practical: A paradigm for designing efficient protocols". In Proceedings of the 1st ACM Conference on Computer and Communications Security (CCC '93). Pages 62–73. (1993).

[103] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. "Random oracles in a quantum world". In Proceedings of the 17th International Conference on the Theory and Application of Cryptology and Information Security. Pages 41–69. (2011).

[104] Seth Lloyd. "Coherent quantum feedback". Physical Review A **62**, 022108 (2000).

[105] John Gough and Matthew R. James. "The series product and its application to quantum feedforward and feedback networks". IEEE Transactions on Automatic Control **54**, 2530–2544 (2009).

[106] Qisheng Wang, Riling Li, and Mingsheng Ying. "Equivalence checking of sequential quantum circuits". IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **41**, 3143–3156 (2022).

[107] Bobak T. Kiani, Giacomo De Palma, Dirk Englund, William Kaminsky, Milad Marvian, and Seth Lloyd. "Quantum advantage for differential equation analysis". Physical Review A **105**, 022415 (2022).

[108] Dominic W. Berry, Andrew M. Childs, Aaron Ostrander, and Guoming Wang. "Quantum algorithm for linear differential equations with exponentially improved dependence on precision". Communications in Mathematical Physics **365**, 1057–1081 (2017).

[109] Mária Kieferová, Artur Scherer, and Dominic W. Berry. "Simulating the dynamics of time-dependent Hamiltonians with a truncated Dyson series". Physical Review A **99**, 042314 (2019).

[110] Dominic W. Berry, Andrew M. Childs, Yuan Su, Xin Wang, and Nathan Wiebe. "Time-dependent Hamiltonian simulation with L^1-norm scaling". Quantum **4**, 254 (2020).

[111] Yi-Hsiang Chen, Amir Kalev, and Itay Hen. "Quantum algorithm for time-dependent Hamiltonian simulation by permutation expansion". PRX Quantum **2**, 030342 (2021).

[112] András Gilyén, Srinivasan Arunachalam, and Nathan Wiebe. "Optimizing quantum optimization algorithms via faster quantum gradient computation". In Proceedings of the 30th Annual ACM SIAM Symposium on Discrete Algorithms (SODA '19). Pages 1425–1444. (2019).

[113] Iordanis Kerenidis and Anupam Prakash. "A quantum interior point method for LPs and SDPs". ACM Transactions on Quantum Computing **1**, 1–32 (2020).

[114] John H. Reif. "Logarithmic depth circuits for algebraic functions". SIAM Journal on Computing **15**, 231–242 (1986).

[115] Mario Szegedy. "Quantum speed-up of Markov chain based algorithms". In Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS '04). Pages 32–41. (2004).

[116] Rolando D. Somma, Gerardo Ortiz, James E. Gubernatis, Emanuel Knill, and Raymond Laflamme. "Simulating physical phenomena by quantum networks". Physical Review A **65**, 042323 (2002).

[117] Iordanis Kerenidis and Anupam Prakash. "Quantum recommendation systems". In 8th Innovations in Theoretical Computer Science Conference (ITCS '17). Volume 67, pages 49:1–49:21. (2017).

[118] Dmitry A. Abanin and Zlatko Papić. "Recent progress in many-body localization". Annalen der Physik **529**, 1700169 (2017).

[119] Fabien Alet and Nicolas Laflorencie. "Many-body localization: An introduction and selected topics". Comptes Rendus Physique **19**, 498–525 (2018).

[120] Philip W. Anderson. "Absence of diffusion in certain random lattices". Physical Review **109**, 1492–1505 (1958).

[121] Dmitry A. Abanin, Ehud Altman, Immanuel Bloch, and Maksym Serbyn. "Colloquium: Many-body localization, thermalization, and entanglement". Reviews of Modern Physics **91**, 021001 (2019).

[122] Joseph Polchinski and Vladimir Rosenhaus. "The spectrum in the Sachdev-Ye-Kitaev model". Journal of High Energy Physics **2016**, 1–25 (2016).

[123] Vladimir Rosenhaus. "An introduction to the SYK model". Journal of Physics A: Mathematical and Theoretical **52**, 323001 (2019).

[124] George E. P. Box and Mervin E. Muller. "A note on the generation of random normal deviates". The Annals of Mathematical Statistics **29**, 610–611 (1958).

[125] Shenglong Xu, Leonard Susskind, Yuan Su, and Brian Swingle. "A sparse model of quantum holography" (2020). arXiv:2008.02303.

[126] Yudong Cao, Jonathan Romero, Jonathan P. Olson, Matthias Degroote, Peter D. Johnson, Mária Kieferová, Ian D. Kivlichan, Tim Menke, Borja Peropadre, Nicolas P. D. Sawaya, Sukin Sim, Libor Veis, and Alán Aspuru-Guzik. "Quantum chemistry in the age of quantum computing". Chemical Reviews **119**, 10856–10915 (2019).

[127] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J. Love, Alán Aspuru-Guzik, and Jeremy L. O'Brien. "A variational eigenvalue solver on a photonic quantum processor". Nature Communications**5** (2014).

[128] Google AI Quantum and Collaborators, Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, Rami Barends, Sergio Boixo, Michael Broughton, Bob B. Buckley, et al. "Hartree-Fock on a superconducting qubit quantum computer". Science **369**, 1084–1089 (2020).

A Details Omitted in Section 3.3

In this appendix we provide details of Section 3.3.

Proof of Lemma 3.25. Similar to the proof of Lemma 3.8, we can write $|\Psi_{j_0}^{(r,m)}\rangle = |\Phi_{j_0}^{(r,m)}\rangle + |\Phi_{j_0}^{(r,m)\perp}\rangle$, where the subnormalized state

$$|\Phi_{j_0}^{(r,m)}\rangle := \frac{1}{\sqrt{(md)^r}} \sum_{\boldsymbol{w}\in[m]^r} \sum_{\boldsymbol{j}\in \mathsf{H}^{\boldsymbol{w}}} |\boldsymbol{w}\rangle\,|\boldsymbol{j}\rangle \otimes \sqrt{\tilde{H}^*_{j_0 j_1} \dots \tilde{H}^*_{j_{r-1} j_r}}\,|\boldsymbol{j}\rangle$$

These subnormalized states also satisfy some orthogonal relations analogous to Lemma 3.7: for all $j,k \in [N]$,

1. $\langle \Phi_j^{(r,m)}|\, S^{(r,m)}\, |\Phi_k^{(r,m)}\rangle = \left(\left(\frac{H}{md}\right)^r\right)_{jk}$.
2. $\langle \Phi_j^{(r,m)}|\, S^{(r,m)}\, |\Phi_k^{(r,m)\perp}\rangle = \langle \Phi_j^{(r,m)\perp}|\, S^{(r,m)}\, |\Phi_k^{(r,m)\perp}\rangle = 0$.

We omit the proof details here, but mention that this can done by the same techniques as in the proof of Lemma 3.7, combined with the fact that $\sum_{w\in[m]} \tilde{H}_{j_s j_{s+1}} = H_{j_s j_{s+1}}$. Using the orthogonal relations the conclusion is easy to obtain as in the proof of Lemma 3.8. □

Proof of Lemma 3.27. The proof is similar to the proof of Lemma 3.14. We verify the conditions in Definition 3.26. Observe that $L(w,j,t) = j \oplus x(w)$ where $\oplus$ is the bit-wise XOR operator.

- We have
$$L^{(r)}(\boldsymbol{w}, j, \boldsymbol{t}) = j \oplus s(w_0) \oplus \dots \oplus s(w_{r-1});$$
that is, take $f(j,k) = j \oplus k$, $g(w,t) = s(w)$ and $\circ$ to be $\oplus$ in (19).
  - f is arithmetic-depth-efficiently computable, while computing $g(w,t)$ requires $O(1)$ queries to $\mathcal{O}_P$.
  - The XOR $\oplus$ is obviously associative and arithmetic-depth-efficiently computable.
- Take the inverse function to be $L^{(-1)}(w,j,k) = s(w)$, which can be computed by a single query to $\mathcal{O}_P$.
- Observe that $(j,k) \in \mathsf{H}_w$ if and only if $j \oplus k = x(w)$; that is, $[\![(j,k) \in \mathsf{H}_w]\!] = [\![j \oplus k = x(w)]\!]$, which is obviously arithmetic-depth-efficiently computable with $O(1)$ queries to $\mathcal{O}_P$.

□

Proof of Lemma 3.28. In this proof we use superscript i to denote the i^{th} bit of a number. We verify the conditions in Definition 3.26. Observe that $L(w,j,t) = j \lhd_{s(w)} \left(t \restriction_{s(w)}\right)$, with $\lhd_s$ and $\restriction_s$ defined in the proof of Lemma 3.15.

- We have
$$L^{(r)}(\boldsymbol{w}, j, \boldsymbol{t}) = j \lhd \left(t_0 \restriction_{s(w_0)}, s(w_0)\right) \lhd^\vee \dots \lhd^\vee \left(t_{r-1} \restriction_{s(w_{r-1})}, s(w_{r-1})\right);$$
that is, take $f(j,(k,x)) = j \lhd (k,x)$, $g(w,t) = \left(t \restriction_{s(w)}, s(w)\right)$, and $\circ$ to be $\lhd^\vee$, where $\lhd$ and $\lhd^\vee$ are defined in the proof of Lemma 3.15.

– As shown in the proof of Lemma 3.15, f is arithmetic-depth-efficiently computable.

The mapping $|w\rangle|0\rangle \mapsto |w\rangle \frac{1}{\sqrt{d}} \sum_{t\in[d]} |g(w,t)\rangle$ can be performed by first querying $\mathcal{O}_P$ to obtain $|s(w)\rangle$ in an ancilla, then conditioned on it implementing n controlled Hadamard gates in parallel, assuming d is a power of two w.l.o.g. This is arithmetic-depth-efficient with $O(1)$ queries.

– $\lhd^\vee$ is associative and arithmetic-depth-efficiently computable, as shown in the proof of Lemma 3.15.

- Take the inverse function to be $L^{(-1)}(w,j,k) = (k \wedge s(w), s(w))$ with $\wedge$ the bit-wise AND operator. This is arithmetic-depth-efficiently computable with $O(1)$ queries to $\mathcal{O}_P$.

- Observe that $(j,k) \in \mathsf{H}_w$ if and only if $j^i = k^i$ for all i with $s(w)^i = 0$; that is,

$$[\![(j,k) \in \mathsf{H}_w]\!] = \bigwedge_{i\in[n]} \left(j^i k^i + (1-j^i)(1-k^i)\right) \cdot \left(1 - s(w)^i\right)$$

which is arithmetic-depth-efficiently computable by an $O(\log n)$-depth and $O(n)$-size quantum circuit by taking $\circ$ to be AND gate in Lemma 2.3, with additional $O(1)$ queries to $\mathcal{O}_P$ to compute $s(w)$.

□

A.1 Proof of Theorem 3.30

Recall that the Hamiltonians considered in Section 3.3 has the form $H = \sum_{w\in[m]} H_w$. To prove Theorem 3.30, following the same line of analysis as in Section 3.2, we will show that:

1. for m-uniform-structured Hamiltonians, the pre-walk, i.e., preparing the pre-walk state $|p_{j_0}^{(r,m)}\rangle$ in (18), can be implemented by a parallel quantum circuit;
2. the re-weight can be efficiently implemented in parallel;
3. the (extended) parallel quantum walk for uniform-structured Hamiltonians can be efficiently implemented in parallel.

Firstly we have the following lemma as a generalization of Corollary 3.19.

Lemma A.1. *For an m-uniform-structured Hamiltonian H, the mapping*

$$|\boldsymbol{w}, j_0, g(\boldsymbol{w},\boldsymbol{t})\rangle \mapsto |\boldsymbol{w}\rangle|\boldsymbol{j}\rangle$$

for all $j_0 \in [N], \boldsymbol{w} \in [m]^r, \boldsymbol{t} \in [d]^r$, where $g(\boldsymbol{w},\boldsymbol{t}) := (g(w_0,t_0),\dots,g(w_{r-1},t_{r-1}))$ and $\boldsymbol{j} \in \mathsf{H}^r$ satisfies $j_{s+1} = L(w_s,j_s,t_s)$ for $s \in [r]$, can be implemented by a quantum circuit of

- *depth $O(1)$ and size $O(r)$ w.r.t. queries to $\mathcal{O}_P$, and*
- *depth $O\left(\log r \cdot \log^2 n\right)$ and size $O(r^2 n^4)$ w.r.t. gates.*

Proof. The lemma can be proved along the same line as Lemma 3.17, Corollary 3.18 and Corollary 3.19. Recall that $m = \operatorname{poly} n$. We omit the proof details here, but mention that only the first two conditions of Definition 3.26 are needed. □

Then the following lemma shows that an (r, m)-pre-walk can be implemented by a parallel quantum circuit, as a generalization of Lemma 3.20.

Lemma A.2 (Pre-walk on m-uniform-structured Hamiltonians)**.** *An (r, m)-pre-walk on an m-uniform-structured Hamiltonian H, i.e., preparing the state $|p_{j_0}^{(r,m)}\rangle$, can be implemented by a quantum circuit of*

- *depth $O(1)$ and size $O(r)$ w.r.t. queries to $\mathcal{O}_P$, and*
- *depth $O\Big(\log r \cdot \log^2 n\Big)$ and size $O(r^2 n^4)$ w.r.t. gates.*

Proof. Let $\mathcal{H}^W \otimes \mathcal{H}^J = \Big(\bigotimes_{s\in[r]} \mathcal{H}_s^W\Big) \otimes \Big(\bigotimes_{s\in[r+1]} \mathcal{H}_s^J\Big)$ be the state space with $\mathcal{H}_s^W = \mathbb{C}^m$ and $\mathcal{H}_s^J = \mathbb{C}^N$. The process of preparing $|p_{j_0}^{(r,m)}\rangle$ is presented below, starting from the initial state $|0\rangle^{\otimes mr} |j_0\rangle |0\rangle^{\otimes Nr}$ with $|j_0\rangle \in \mathbb{C}^N, |0\rangle \in \mathbb{C}$.

1. Prepare a superposition over computational basis states of size m^r in the subspace $\mathcal{H}^W$:
$$\frac{1}{\sqrt{m^r}} \sum_{\boldsymbol{w}\in[m]^r} |\boldsymbol{w}\rangle |j_0\rangle |0\rangle^{\otimes Nr}.$$
2. For each $s \in [r]$, perform in parallel the mapping $|w_s\rangle |0\rangle \mapsto |w_s\rangle \frac{1}{\sqrt{d}} \sum_{t_s\in[d]} |g(w_s, t_s)\rangle$ in the subspace $\mathcal{H}_s^W \otimes \mathcal{H}_{s+1}^J$, to obtain the state
$$\frac{1}{\sqrt{(md)^r}} \sum_{\boldsymbol{w}\in[m]^r} \sum_{\boldsymbol{t}\in[d]^r} |\boldsymbol{w}\rangle |j_0\rangle |g(\boldsymbol{w}, \boldsymbol{t})\rangle$$
3. Apply Lemma A.1 to obtain the goal state $|p_{j_0}^{(r,m)}\rangle = \frac{1}{\sqrt{(md)^r}} \sum_{\boldsymbol{w}\in[m]^r} \sum_{\boldsymbol{j}\in \mathsf{H}^{\boldsymbol{w}}} |\boldsymbol{w}\rangle |\boldsymbol{j}\rangle$.

Step 1 can be done by an $O(1)$-depth and $O(r \log m)$-size quantum circuit using Hadamard gates in parallel, assuming m is a power of two w.l.o.g. Each mapping
$$|w\rangle |0\rangle \mapsto |w\rangle \frac{1}{\sqrt{d^r}} \sum_{t\in[d]} |g(w, t)\rangle$$
in Step 2 is arithmetic-depth-efficient with $O(1)$-queries to $\mathcal{O}_P$ due to Definition 3.26. Recall that $m = \operatorname{poly} n$, the final complexity follows from summing up these complexities combined with Lemma A.1. □

Now we move to the re-weight analysis.

Lemma A.3. *Re-weight of $|p_{j_0}^{(r,m)}\rangle$, i.e., performing the mapping $|p_{j_0}^{(r,m)}\rangle \mapsto |\Psi_{j_0}^{(r,m)}\rangle$, where $|\Psi_{j_0}^{(r,m)}\rangle$ is defined in (17), can be implemented to precision ϵ by a quantum circuit of*

- *depth $O(1)$ and size $O(r)$ w.r.t. queries to $\mathcal{O}_H^b$ with $b = O(\log(m/\epsilon))$,*
- *depth $O(1)$ and size $O(rm)$ w.r.t. queries to $\mathcal{O}_P$, and*
- *depth $O\Big(\log^2 n + \log^2 \log(m/\epsilon)\Big)$ and size $O\Big(r\Big[mn^4 + \log^4(m/\epsilon)\Big]\Big)$ w.r.t. gates,*

for $r = \operatorname{polylog}(1/\epsilon)$.

Proof. The analysis is exactly the same as in Lemma 3.21, with additional complexities of:

1. Computing $c(j,k) = \sum_{w\in[m]} [\![(j,k) \in \mathsf{H}_w]\!]$ in r subspaces, each of which can be implemented by a quantum circuit of

   - depth $O(1)$ and size $O(m)$ w.r.t. queries to $\mathcal{O}_P$, and
   - depth $O\left(\log^2 n\right)$ and size $O(mn^4)$ w.r.t. gates.

   To see this, note that for each subspace, one can first creates m copies of (j,k) by COPY_m^{2n}, then compute each $[\![(j,k) \in \mathsf{H}_w]\!]$ arithmetic-depth-efficiently with $O(1)$ queries to $\mathcal{O}_P$ for $w \in [m]$ in parallel, due to the third condition in Definition 3.26. Finally apply Corollary 2.4 to compute $c(j,k)$ in each subspace, followed by garbage cleaning by reverse computation. The final complexity follows from the assumption $m = \operatorname{poly} n$.

2. Scaled total precision $\tilde{\epsilon} = \epsilon/m$ for the aforementioned construction of the oracle $\mathcal{O}_{\tilde{H}}$.

3. Arithmetic-depth-efficient circuits for the division $\tilde{H}_{jk} = H_{jk}/c(j,k)$ by Lemma 2.6.

□

Finally, combining Lemma A.2, Lemma A.3 and the negligible complexity of implementing $S^{(r,m)}$ gives the complexity of the (r,m)-parallel quantum walk $Q^{(r,m)}$ for m-uniform-structured Hamiltonians in Theorem 3.30.

B Definition of State Preparation

In this appendix we show that the state preparation unitary in Definition 4.2 is a special case of Definition 51 in [36]:

Definition B.1 (State preparation pair [36])**.** Let $\boldsymbol{a} \in \mathbb{C}^R$ and $\|\boldsymbol{a}\|_1 \le \alpha$, the pair of unitaries (V_C, V_D) is called an (α, b, ϵ)-state-preparation-pair of $\boldsymbol{a}$ if $V_C|0\rangle = \sum_{r\in[2^b]} c_r|r\rangle$ and $V_D|0\rangle = \sum_{r\in[2^b]} d_r|r\rangle$ with $|0\rangle \in \mathbb{C}^{2^b}$, such that $\sum_{r\in[R]} |\alpha(c_r^* d_r) - a_r| \le \epsilon$ and for all $r \in R, \ldots, 2^b - 1$ we have $c_r^* d_r = 0$.

This is shown more precisely in the following lemma.

Lemma B.2. *Let $\boldsymbol{a} \in \mathbb{C}^R$ and $\alpha := \|\boldsymbol{a}\|_1$, where we assume $R = 2^s \ge 4$ w.l.o.g. Let V be an (α,ϵ)-state-preparation-unitary of $\boldsymbol{a}$, then $(V^\dagger, V)$ is an $(\alpha, s, \alpha R\epsilon)$-state-preparation-pair of $\boldsymbol{a}$.*

Proof. Let $V|0\rangle = \sum_{r\in[R]} \sqrt{v_r}|r\rangle$. By Definition 4.2, we have

$$\sqrt{\sum_{r\in[R]} \left|\sqrt{v_r} - \sqrt{\frac{a_r}{\alpha}}\right|^2} \le \epsilon. \tag{31}$$

Note that

$$\sum_{r\in[R]} \left|v_r - \frac{a_r}{\alpha}\right| \le \sum_{r\in[R]} \left|\sqrt{v_r} - \sqrt{\frac{a_r}{\alpha}}\right| \cdot \left|\sqrt{v_r} + \sqrt{\frac{a_r}{\alpha}}\right| \le 2 \sum_{r\in[R]} \left|\sqrt{v_r} - \sqrt{\frac{a_r}{\alpha}}\right|. \tag{32}$$

By the well-known inequality between l_1-norm and l_2-norm: $\|\boldsymbol{x}\|_1 \leq \sqrt{R}\cdot\|\boldsymbol{x}\|_2$ for $\boldsymbol{x} \in \mathbb{C}^R$, the RHS of (32) is upper bounded by

$$2\sqrt{R}\cdot\sqrt{\sum_{r\in[R]}\left|\sqrt{v_r}-\sqrt{\frac{a_r}{\alpha}}\right|^2} \leq 2\sqrt{R}\epsilon \leq R\epsilon,$$

where the first inequality comes from (31), and the second inequality uses the assumption $R \geq 4$. Thus we have $\sum_{r\in[R]}|v_r - \frac{a_r}{\alpha}| \leq R\epsilon$. Now take $b = s$, $V_C = V^\dagger$ and $V_D = V$ in Definition B.1, it immediately follows that $\sum_{r\in[R]}|\alpha(c_r^* d_r) - a_r| = \alpha\sum_{r\in[R]}|v_r - \frac{a_r}{\alpha}| \leq \alpha R\epsilon$, which implies that $\left(V^\dagger, V\right)$ is an $(\alpha, s, \alpha R\epsilon)$-state-preparation-pair of $\boldsymbol{a}$. □

We note that in the above lemma, the upper bound $\alpha R\epsilon$ on the precision of the state preparation pair is very loose, which is for simplicity of the proofs in Section 4 with the final results unchanged.