11

Hamiltonian simulation

The task of Hamiltonian simulation is to approximately compile the evolution under a Hamiltonian $H(t)$, for time t, into a sequence of quantum gates. For a time-independent Hamiltonian, solving the Schrödinger equation (setting $\hbar = 1$) yields a time evolution operator $U(t) = e^{-iHt}$. In this chapter, we will discuss the equivalent operator $U(t) = e^{iHt}$, which is the more common definition in an algorithmic setting. We will assume $t \geq 0$, without loss of generality. The Hamiltonian of interest can arise from physical systems (e.g., quantum chemistry, condensed matter systems, or quantum field theories) but may also be constructed for other applications, such as differential equation simulation. Quantum simulation does not give full access to the amplitudes of the wavefunction during the simulation, unlike classical approaches based on exact diagonalization (or similar methods). Instead, we are only able to measure observables with respect to the time-evolved state, or use the state as an input to other quantum subroutines. Nevertheless, there are no known efficient classical methods that achieve this for general local or sparse Hamiltonians, suggesting an exponential quantum speedup. In fact, as a quantum computation can be expressed as a time evolution under a sequence of local (time-dependent) Hamiltonians, quantum simulation (i.e., time evolution and measurement of a given observable) is a BQP-complete problem.

Hamiltonian simulation algorithms require access to the Hamiltonian. There are three commonly used input models. The Pauli input model assumes that the Hamiltonian is given classically as a sum of products of Pauli operators, for example, $H = \sum_l h_l H_l$, where h_l are coefficients and H_l are multiqubit Pauli products. The d-sparse access model assumes that the Hamiltonian is a sparse matrix with at most d nonzero elements per row or column. We require that the locations of the nonzero elements and their values are efficient to compute classically. The density matrix access model assumes that the Hamiltonian corresponds to a density matrix, which we are either provided access to

copies of [708], or given a unitary that prepares a purification of the density matrix [717]. All of these input models can be used to prepare block-encodings of the Hamiltonian, which provides a standard-form access model that generalizes the above input models. Block-encodings are the input model of choice for some algorithms for Hamiltonian simulation (e.g., qubitization with quantum signal processing) [717].

Hamiltonian simulation can be used as a subroutine in a range of algorithms, including quantum phase estimation, quantum linear system solvers, Gibbs state preparation, and the quantum adiabatic algorithm. We remark that some of these algorithms are implicitly using Hamiltonian simulation to provide coherent, unitary access to the Hamiltonian. This can be particularly useful if few ancilla qubits are available, which may inhibit the use of some approaches to coherently access the Hamiltonian (e.g., block-encodings based on linear combinations of unitaries) but does not prevent the use of Hamiltonian simulation based on product formulas.

Each algorithm has its own advantages and disadvantages, as described at a high level in Table 11.1. Specific optimizations of each algorithm may be available for a given Hamiltonian. One can also consider hybridized methods combining two or more of the algorithms [718, 720, 820, 484, 852, 1027]. There are also other methods for Hamiltonian simulation, such as quantum walks [275, 133, 136] or density matrix–based Hamiltonian simulation [708, 619], which we do not discuss due to their less widespread use as algorithmic primitives for the applications discussed elsewhere in this book.

The authors are grateful to Yuan Su for reviewing this chapter.

11.1 Product formulas

Rough overview (in words)

Product formulas (or Trotter–Suzuki formulas/Trotterization) [705] are the most commonly used approach for Hamiltonian simulation, and are applicable to Hamiltonians in the Pauli access model and the sparse access model (see below for definitions of these models). Product formulas divide the evolution into a repeating sequence of short unitary evolutions under subterms of the Hamiltonian. These subterm evolutions have a known decomposition into elementary quantum gates. The error in product formulas depends on the commutators between different terms in the decomposition; if all of the terms in the Hamiltonian commute, product formulas are exact.

	Product formula (order k)	qDRIFT	Taylor and Dyson series	QSP/QSVT
# Qubits	$O(n)$	$O(n)$	$O(n+\log(L))$	$O(n+\log(L))$
Access model	Pauli Sparse	Pauli Sparse	Pauli Sparse Block-encoding	Block-encoding
Scaling	$O\left(5^{2k} nL\|\|H\|\|_1 t\left(\|\|H\|\|_1 t\epsilon^{-1}\right)^{\frac{1}{2k}}\right)^a$	$O\left(n\|\|H\|\|_1^2 t^2 \epsilon^{-1}\right)$	$\widetilde{O}\left(\|\|H\|\|_1 tnL\log(\epsilon^{-1})\right)$	$O\left(nL(\|\|H\|\|_1 t+\log(\epsilon^{-1}))\right)^b$
Pros	Commutator scaling. Simple implementation. Empirical performance. Minimal ancilla qubits.	L-independent scaling. No ancilla qubits.	$\log(1/\epsilon)$ scaling. Time-dependent simulations.	Optimal scaling with t, ϵ. Few ancilla qubits for algorithm.
Cons	Scaling with t, ϵ at low orders. Exponential prefactor (in order k).	Scaling with t, ϵ.	Many ancilla qubits if using noncompressed variant [718].	Time-dependent simulation. Ancilla/gate cost of block-encoding.

Table 11.1 High-level comparison of Hamiltonian simulation techniques. Quantitative comparisons assume a Pauli input model (which can easily be used to prepare a block-encoding of the Hamiltonian). For the stated complexity, we consider evolution $U(t) = e^{iHt}$ for time t under a time-independent Hamiltonian H on n qubits, given as a sum of L Pauli products $H = \sum_{j=1}^{L} h_j P_j$. The evolution is approximate to error ϵ in the spectral norm (diamond norm for qDRIFT). We define $\|H\|_1 = \sum_{j=1}^{L} |h_j|$. The qubit requirement for the Taylor and Dyson series method omits additional additive factors that scale logarithmically with the norm and/or derivative of the Hamiltonian. In specific applications it may be possible to reduce the number of qubits and/or gate complexity further by exploiting knowledge of the system, such as symmetries, commutation structure, or energy scales. For example, the factor of n present in the above complexities may be reduced by exploiting locality in the Pauli product terms of the Hamiltonian.

[a] The factor of n can be reduced to w when each Pauli term P_j acts nontrivially on at most w sites. The factor $\|H\|_1^{1+1/2k}$ can be reduced by exploiting commutativity of the various P_j.

[b] The factor nL derives from an upper bound on the gate complexity of block-encoding, and it can often be significantly improved by exploiting structure in h_j and H_j.

Product formula approaches have also been extended to treat time-dependent Hamiltonians [556, 1038, 1031, 34, 839]. In the following discussion, we will restrict our focus to the time-independent case, noting that the time-dependent approaches are executed in the same way, but have a slightly more complex error analysis.

Rough overview (in math)

Given a Hamiltonian H, desired evolution time t, and error ϵ, return a circuit $U(t)$ made of elementary gates such that

$$\|U(t) - \mathrm{e}^{\mathrm{i}Ht}\| \leq \epsilon.$$

In the above, we use the operator norm $\|\cdot\|$ (the maximal singular value) to quantify the quality of approximation, which controls the error for arbitrary input states (in trace distance) and for observables. This worst-case metric is mathematically convenient, but, as discussed below, tighter bounds may be obtained by using error metrics more closely aligned with the specification of the problem.

A product formula generates $U(t)$ through a product of easy-to-implement evolutions under terms in the Hamiltonian. For a Hamiltonian decomposition $H = \sum_{j=1}^{L} H_j$ with L terms, the first-order product formula with r steps is

$$S_1(t) = \left(\prod\nolimits_{j=1}^{L} \mathrm{e}^{\mathrm{i}H_j t/r}\right)^r.$$

The error in the first-order product formula is upper bounded as [286]

$$\|S_1(t) - \mathrm{e}^{\mathrm{i}Ht}\| \leq \frac{t^2}{2r} \sum_{i=1}^{L} \left\| \sum_{j=i+1}^{L} [H_i, H_j] \right\| \leq \frac{\|H\|_1^2 t^2}{2r},$$

where $\|H\|_1 = \sum_{j=1}^{L} \|H_j\|$. Higher-order formulas can be defined recursively, and are referred to as $(2k)$th-order product formulas. The error in a recursively defined $(2k)$th-order product formula is bounded by [286]

$$\|S_{2k}(t) - \mathrm{e}^{\mathrm{i}Ht}\| = O\left(\frac{\|H\|_1^{2k+1} t^{2k+1}}{r^{2k}}\right).$$

Product formulas can be applied to d-sparse Hamiltonians (at most d nonzero elements per row/column) with efficiently row computable nonzero elements [11]. Access to the nonzero elements of the Hamiltonian is provided via oracles O_f and O_H. The oracle O_f returns the column index (j) of the $k \in \{1, \ldots, d\}$th nonzero element in row i. The oracle O_H returns the value of

the matrix element H_{ij}.

$$O_f : O_f|k\rangle|i\rangle|0\rangle = |k\rangle|i\rangle|j\rangle$$
$$O_H : O_H|i\rangle|j\rangle|0\rangle = |i\rangle|j\rangle|H_{ij}\rangle.$$

Using graph-coloring algorithms, a d-sparse Hamiltonian H can be efficiently decomposed into a sum of efficiently simulable sparser Hamiltonians [134, 278]. Illustrating this idea with the approach of [134], we can decompose $H = \sum_{j=1}^{6d^2} H_j$, where each H_j is 1-sparse. The nonzero elements of a given H_j can be computed coherently using $O(\log^*(n))$ queries to O_f, O_H, where $\log^*$ is the iterated logarithm.[1] Time evolution under a 1-sparse Hamiltonian can be implemented efficiently using the approach of [11]. To simulate e^{iHt} using, for example, a first-order product formula, we sequentially apply each $e^{iH_j t}$ using the methods outlined above.

As a special case of the d-sparse access model, one can consider Hamiltonians given as a linear combination of L Pauli terms $H = \sum_{j=1}^{L} H_j = \sum_{j=1}^{L} \alpha_j P_j$, as each Pauli tensor product is already a 1-sparse matrix (so in this case, $d \leq L$). Time evolution under each Pauli term (or in some cases, groups of Pauli terms) can be simulated efficiently (see, e.g., [705, 801]), considerably simplifying the d-sparse construction by removing the need for oracles O_f and O_H.

Dominant resource cost (gates/qubits)

For an n-qubit Hamiltonian, product formulas act on n qubits. In the Pauli access model, no additional ancilla qubits are required. In the sparse access model, ancilla qubits may be required to implement the oracles O_f and O_H and to implement time evolution under 1-sparse Hamiltonians H_j.

The gate complexity is obtained by choosing the number of Trotter steps r sufficiently large to obtain an error ϵ and multiplying by the complexity of implementing each step of the product formula. It is necessary to balance the improved asymptotic scaling with t and ϵ of higher-order Trotter formulas against the exponentially growing prefactor of the higher-order formulas. In practical simulations of chemistry, condensed matter systems, or quantum field theories, a low-order formula (2nd–6th) typically minimizes the gate count.

A recursively defined $(2k)$th-order product formula (i.e., the first-order formula is given by $k = 1/2$, and is the base case) for simulating a d-sparse Hamiltonian for time t to accuracy ϵ requires [278]

$$O\left(5^{2k} d^2 (d + \log^* n) \|H\| t \left(\frac{d\|H\|t}{\epsilon}\right)^{1/2k}\right)$$

[1] For practical purposes, the iterated logarithm is essentially constant, since $\log^*(n) \leq 5$ for all $n \leq 2^{65536}$.

calls to the oracles O_f and O_H.

A recursively defined $(2k)$th-order product formula for simulating an L-term Hamiltonian in the Pauli access model for time t to accuracy ϵ requires [286]

$$O\left(5^{2k} nLt\left(\frac{t\alpha_{\text{comm},k}}{\epsilon}\right)^{1/2k}\right) \tag{11.1}$$

elementary single- and two-qubit gates, where

$$\alpha_{\text{comm},k} = \sum_{i_1,i_2,\ldots,i_{2k+1}} \|[H_{i_{2k+1}}, \ldots [H_{i_2}, H_{i_1}]]\| .$$

The dependence on $\alpha_{\text{comm},k}$ can be tightened and calculated for lower-order formulas (see [286] for full calculations). The dependence on n can be reduced to w for local Hamiltonians with Pauli terms that each act on at most w qubits.

Caveats

The error bounds of product formulas in the Pauli access model have been the object of significant investigation. Evaluating the tightest spectral norm bounds requires computing a large number of commutators between the terms in the Hamiltonian, which can be computationally intensive. Numerical simulations have shown that the commutator bounds can be loose by several orders of magnitude for chemical [72, 840] or spin [283] systems.

The spectral norm is the worst-case metric; it is an active area of research to find error metrics better suited to the problem at hand. For example, one may consider the *average*-case error over random input states [257, 1087] by the normalized Frobenius norm $\|U(t) - e^{iHt}\|_F / \sqrt{2^n}$. Recently, in [257] it was shown that the average-case error can be much smaller than the worst-case error for systems with large connectivity. More directly, one can also compute the Trotter error associated with input states from the low-energy [891, 516] or low-particle-number subspace [991, 963].

The gate counts of product formula approaches can also be reduced by grouping together mutually commuting terms such that they can be implemented using fewer gates than would be required to implement all the terms individually [124, 630, 225]. One can also reduce the number of Trotter steps required by randomizing the ordering of the terms [284, 288, 839] (although this must be balanced against any compilation benefits that may be obtained from a fixed ordering).

Example use cases

- Physical systems simulation: quantum chemistry, condensed matter systems, quantum field theories.

- Algorithms: quantum phase estimation, quantum linear system solvers, Gibbs state preparation, quantum adiabatic algorithm.

Further reading

- A rigorous derivation of the error in product formulas [286].
- A comparison of product formula methods with other approaches to Hamiltonian simulation for a concrete problem of interest [283].

11.2 qDRIFT

Rough overview (in words)

The *quantum stochastic drift protocol* [222], abbreviated as qDRIFT, operates in the Pauli access model[2] and approximates the Hamiltonian simulation channel (as opposed to the unitary) by randomly sampling a term from the Hamiltonian (according to the coefficient magnitudes), and then evolving under the chosen term. This process is repeated for a number of steps. Because it approximates the channel, rather than the unitary, it can be more difficult to use qDRIFT as a coherent subroutine in other algorithms (see §Caveats below).

The error in qDRIFT depends on the 1-norm of Hamiltonian coefficients. One main advantage of qDRIFT is that it does not explicitly depend on the number of terms in the Hamiltonian and has small constant overheads, making it well suited to systems with rapidly decaying interaction strengths, dominated by a few large terms. However, qDRIFT's time and error dependence are asymptotically worse than other methods, which seems to originate from the algorithm's randomized nature [258]. qDRIFT can also be extended to time-dependent Hamiltonian simulation with a Hamiltonian $H(t)$, where the gate count of the algorithm scales as $\int_0^t \|H(t')\| \mathrm{d}t'$, rather than as $t \max_{t'} \|H(t')\|$ like other Hamiltonian simulation algorithms [141]. We will restrict our discussion below to the time-independent case.

Rough overview (in math)

Given a Hamiltonian in the Pauli decomposition $H = \sum_i h_i H_i$ (with $\|H_i\| = 1$), qDRIFT provides a stochastic channel $\mathcal{N}$ that, when applied for N steps, approximates the Hamiltonian simulation channel

$$\|\mathcal{N}^N - \mathrm{e}^{\mathrm{i}Ht}(\cdot)\mathrm{e}^{-\mathrm{i}Ht}\|_\diamond \leq \epsilon$$

[2] qDRIFT was originally formulated, and is typically presented, for the Pauli access model [222], but the algorithm appears compatible with the d-sparse access model by applying it to the d-sparse decompositions in [134, 278] [961].

to within diamond-norm error ϵ.

qDRIFT proceeds by randomly sampling terms according to their relative importance

$$X_k \overset{i.i.d.}{\sim} \frac{h_i H_i}{p_i}, \quad \text{where} \quad p_i = \frac{|h_i|}{\|H\|_1}$$

and $\|H\|_1 := \sum_i |h_i|$ is the sum of the strengths. Each step of qDRIFT then evolves the randomly sampled term X_k for a short period of time t/N, where N is a free parameter determining the number of qDRIFT steps, which controls the error in the simulation. The result is the following quantum channel:

$$\mathcal{N}[\rho] = \mathbb{E}[e^{i(t/N)X_k} \rho e^{-i(t/N)X_k}].$$

As discussed above, this channel is repeated for N steps, in order to approximate the Hamiltonian simulation channel.

Dominant resource cost (gates/qubits)

For an n-qubit Hamiltonian, qDRIFT acts on n register qubits, and no additional ancilla qubits are required.

In order to simulate the Hamiltonian evolution channel to within diamond-norm error ϵ, we require

$$N = O\left(\frac{\|H\|_1^2 t^2}{\epsilon}\right)$$

steps of qDRIFT [222, 258]. While the diamond-norm is a different error metric to the spectral norm used in other articles in this section, both metrics provide upper bounds on the error in an observable measured with respect to the time-evolved state [222]. For unitary channels, the diamond norm is effectively equal to the spectral norm (see, e.g., discussion in [480], up to constant factors).

The gate complexity is the number of steps multiplied by the individual costs of the elementary evolution $e^{i(t/N)X_k}$, which scales linearly with the locality of the Pauli operator X_k. When using qDRIFT to time evolve a state (e.g., for the purpose of measuring an observable), it is important to average the results over a sufficient number of independently sampled qDRIFT circuits [222].

Caveats

The qDRIFT algorithm has a quadratic dependence on time and a linear dependence on the inverse error $1/\epsilon$, while other Hamiltonian simulation methods can achieve linear time dependence and logarithmic inverse error dependence. A higher-order variant of qDRIFT was recently developed that improves the error dependence, but it is only suitable for estimating the expectation value of

observables with respect to the time-evolved state, rather than approximating the unitary channel itself [794]. It is currently unclear how to design higher-order variants of qDRIFT that improve the time dependence, which appears to result from the randomized nature of the algorithm [258].

As discussed above, qDRIFT approximates the time evolution channel, rather than the unitary e^{iHt}. As a result, it can be difficult to incorporate as a subroutine in algorithms that seek to manipulate the unitary directly—for example, measuring $\mathrm{Tr}(U(t)\rho)$. Tasks of this form feature in some approaches for phase estimation [690], motivating alternate, qDRIFT-inspired approaches, in order to exploit qDRIFT-like benefits [1013].

Example use cases

- Physical systems simulation: quantum chemistry, condensed matter systems, quantum field theories.
- Algorithms: quantum phase estimation, quantum linear system solvers, Gibbs state preparation, quantum adiabatic algorithm.
- Hybridization with other quantum simulation methods [820, 852, 484].
- Using importance sampling to incorporate variable gate costs for simulating different terms X_k [621].

11.3 Taylor and Dyson series (linear combination of unitaries)

Rough overview (in words)

Taylor and Dyson series approaches for Hamiltonian simulation expand the time evolution operator as a Taylor series (time independent) [137] or Dyson series (time dependent) [615, 141] and use the linear combination of unitaries (LCU) primitive to apply the terms in the expansion, followed by (robust, oblivious) amplitude amplification to boost the success probability close to unity. These methods are close to being asymptotically optimal, achieving linear scaling in time and logarithmic dependence on the error. However, they use a large number of ancilla qubits, compared to other Hamiltonian simulation algorithms.

Rough overview (in math)

We focus on the time-independent case and follow the presentation in [137]. Given a Hamiltonian H, desired evolution time t, and error ϵ, return a circuit

$U(t)$ made of elementary gates such that

$$\|U(t) - \mathrm{e}^{\mathrm{i}Ht}\| \leq \epsilon.$$

In the above, we use the operator norm (the maximal singular value) to quantify the worst-case error in the simulation.

The total evolution time t is divided into r segments. In each segment, we evolve under an approximation of $\mathrm{e}^{\mathrm{i}Ht/r}$. The Hamiltonian is decomposed into a linear combination of unitary operations $H = \sum_{l=1}^{L} \alpha_l H_l$, where we choose α_l real and positive by shifting phases into H_l, and $\|H_l\| = 1$. This decomposition appears naturally when the Hamiltonian is given as a linear combination of Pauli products. We approximate $\mathrm{e}^{\mathrm{i}Ht/r}$ using a Taylor expansion truncated to degree K

$$\begin{aligned}\mathrm{e}^{\mathrm{i}Ht/r} \approx U(t/r) &:= \sum_{k=0}^{K} \frac{1}{k!} (\mathrm{i}Ht/r)^k \\ &= \sum_{k=0}^{K} \sum_{l_1,\ldots,l_k=1}^{L} \frac{(\mathrm{i}t/r)^k}{k!} \alpha_{l_1} \ldots \alpha_{l_k} H_{l_1} \ldots H_{l_k}.\end{aligned}$$

Each segment $U(t/r)$ is implemented using robust oblivious amplitude amplification. Amplitude amplification is necessary because truncating the Taylor series at degree K makes $U(t/r)$ non-unitary. However, textbook amplitude amplification necessitates reflecting around the initial state (as well as the "good" state), which would be problematic since Hamiltonian simulation requires synthesizing a unitary that works simultaneously for all input states. This issue can be circumvented using oblivious amplitude amplification: we are given a unitary V such that for any state $|\psi\rangle$, we have $V|0^m\rangle|\psi\rangle = a|0^m\rangle U|\psi\rangle + b|0_\perp^m \phi\rangle$, for a unitary operator U, and the goal is to amplify the state $|0^m\rangle U|\psi\rangle$ to be obtained with probability 1 (we can recognize V as an $(a, m, 0)$ unitary block-encoding of U). A further problem is that the above operator $U(t/r)$ is non-unitary, and so deviates from the formulation of oblivious amplitude amplification [135]. The proven "robustness" property of oblivious amplitude amplification [137] ensures that the error induced by treating $U(t/r)$ as a probabilistically implemented unitary does not accumulate.

The value of K controls the error in the simulation and can be chosen as

$$K = O\left(\frac{\log(\|H\|_1 t/\epsilon)}{\log\log(\|H\|_1 t/\epsilon)}\right),$$

where we define $\|H\|_1 := \sum_{l=1}^{L} \alpha_l$. The total time evolution is divided into $r = \lceil \|H\|_1 t/\ln(2) \rceil$ segments, each of duration $\ln(2)/\|H\|_1$, which ensures that a

single application of robust oblivious amplitude amplification boosts the success probability of the segment to unity.

Within each segment, we apply $U(t/r)$ using the LCU primitive. This technique can be applied to Hamiltonians given in both the Pauli and d-sparse access models. For the Pauli access model, the Hamiltonian is already in the form of a linear combination of unitary operators. For the d-sparse case, we can use graph coloring algorithms [134, 278] to decompose the d-sparse Hamiltonian into a linear combination of unitaries, where each unitary is 1-sparse and self-inverse.

Dominant resource cost (gates/qubits)

In addition to the n-qubit data register, the Taylor series approach requires a number of ancilla registers to implement the LCU technique. In the original formulation [137], a register with K qubits is used to control the degree of the Taylor expansion, storing the value as $|k\rangle = |1^{\otimes k}0^{\otimes(K-k)}\rangle$. An additional K registers, each containing $\lceil \log_2(L) \rceil$ qubits, are used to index the possible values of each of the possible H_{l_k}. Hence, the overall space complexity of the original formulation [137] is $O(n + K\log(L)) = O(n + \log(\|H\|_1 t/\epsilon)\log(L))$. In [718] it was shown how to reduce the space complexity to $O(n + \log(K) + \log(L))$ using quantum counter circuits.

Additional ancilla qubits may be required to implement the LCU gadget (e.g., in the sparse access model) or for the reflections used in robust oblivious amplitude amplification.

As discussed above, implementing each segment requires one use of robust oblivious amplitude amplification, which makes two calls to the LCU circuit and one call to its inverse. The method incurs approximation errors from truncating the Taylor series at degree K and from the use of robust oblivious amplitude amplification. The resulting error per segment is bounded by $(e\ln(2)/(K+1))^{K+1}$.

The cost of the LCU circuit depends on the Hamiltonian access model. For the case of the Pauli access model, the LCU circuit requires two calls to a PREPARE operation that prepares the ancilla registers with the correct coefficients. In the compressed formulation [718], this requires $O(L + K)$ gates (compared to $O(LK)$ gates in the original formulation [137]). The LCU circuit also requires one call to a SELECT oracle, which can be implemented using K controlled select(H) operations. Each of these K operations can be implemented using $O(Ln)$ elementary gates [283, 75] (using quantum read-only memory).

The overall gate complexity in the Pauli access model is thus

$$O\left(\frac{\|H\|_1 tLn \log(\|H\|_1 t/\epsilon)}{\log\log(\|H\|_1 t/\epsilon)}\right) = \widetilde{O}\left(\|H\|_1 tLn \log\left(\frac{1}{\epsilon}\right)\right).$$

Using the LCU approach applied to a 1-sparse decomposition of a d-sparse Hamiltonian, the overall complexity is [137]

$$O\left(\frac{d^2\|H\|_{\max} tn \log^2(d^2|H|_{\max} t/\epsilon)}{\log\log(d^2\|H\|_{\max} t/\epsilon)}\right) = \widetilde{O}\left(d^2\|H\|_{\max} tn \log^2\left(\frac{1}{\epsilon}\right)\right),$$

where $\|H\|_{\max} = \max_{i,j}|\langle i|H|j\rangle|$. Using the amplification technique of [715], which utilizes quantum singular value transformation (QSVT), the dependence on d and $\|H\|_{\max}$ can be improved in some cases.

The extension to time-dependent Hamiltonians, through the use of a Dyson series, requires an additional "clock" register to store the time value and introduces a logarithmic dependence on the time derivative of the Hamiltonian [615, 141, 718].

Caveats

Concrete resource estimates for physical systems of interest have observed that the Taylor series approach may require more ancilla qubits and gates than product formulas or quantum signal processing approaches for Hamiltonian simulation [283], although these qubit counts would be improved by the subsequent compression approach to the algorithm [718]. The gate complexity of the algorithm can be reduced by exploiting anticommutativity in the Hamiltonian [1086], adding a corrective operation [804], or pruning terms with small magnitudes from the expansion [759].

Example use cases

- Physical systems simulation: quantum chemistry (see [73, 74, 962, 718]), condensed matter systems, quantum field theories.
- Algorithms: quantum phase estimation, quantum linear system solvers, Gibbs state preparation, quantum adiabatic algorithm.
- Hamiltonian simulation in the interaction picture [718].

Further reading

- A comparison of several Hamiltonian simulation algorithms, including Taylor series [283].
- Derivations of the compressed variants of Hamiltonian simulation via Taylor/Dyson series [718, Appendices B & D].

11.4 Quantum signal processing / quantum singular value transformation

Rough overview (in words)

Quantum signal processing (QSP) and quantum singular value transformation (QSVT) are techniques for applying polynomial transformations to block-encoded operators. These techniques can be used to implement Hamiltonian simulation, given a block-encoding of the Hamiltonian. Both approaches have optimal scaling with t and ϵ for time-independent Hamiltonians.

QSP was initially developed for the d-sparse access model [716]. Through the introduction of block-encodings and qubitization, it was made applicable in a standard form to Hamiltonians in a Pauli access model, d-sparse access model, or given as density matrices (where we are given access to a unitary that prepares a purification of the density matrix) [717]. QSVT was later developed as a more general and direct route to the results of QSP [429].

Hamiltonian simulation via QSP or QSVT is less well suited to time-dependent Hamiltonians, as the need to Trotterize the time-dependent evolution breaks the optimal dependence on the parameters.

Rough overview (in math)

Access to the Hamiltonian H is provided by an $(\alpha, m, 0)$-block-encoding U_H (the case of approximate block-encodings can be treated using [429, Lemma 22]) such that

$$(\langle 0^m| \otimes I) U_H (|0^m\rangle \otimes I) = H/\alpha.$$

The Hamiltonian has a spectral decomposition of $\sum_\lambda \lambda |\lambda\rangle\langle\lambda|$. We seek to use U_H to implement an operator $U(t)$ approximating

$$\|U(t) - \sum_\lambda \mathrm{e}^{\mathrm{i}\lambda t} |\lambda\rangle\langle\lambda|\| \leq \epsilon.$$

Qubitization converts U_H into a more structured unitary W (which is also a block-encoding of the Hamiltonian). The eigenvalues of W are $\mathrm{e}^{\pm \mathrm{i} \arccos(\lambda/\alpha)}$, directly related to those of H. QSP then enables polynomial transformations to be applied to these eigenvalues, which defines the application of the polynomial to W. This concept can be generalized via QSVT, which effectively unifies the qubitization and QSP step.

In both cases, our goal is to implement a block-encoding of $U(t) \approx \sum_\lambda \mathrm{e}^{\mathrm{i}\lambda t} |\lambda\rangle\langle\lambda|$, which defines Hamiltonian simulation. In QSVT we separately implement polynomials approximating $\cos(\lambda t)$ and $i \sin(\lambda t)$, combine them using a linear combination of block-encodings, and boost the success probability using three-step oblivious amplitude amplification. Further details

can be found in [429, 744]. Meanwhile, QSP implements $\exp(itH)$ directly but requires an additional ancilla qubit and controlled access to a Hermitian block-encoding U'_H, which, when implemented via Eq. (10.5), uses both controlled U_H and $U_H^\dagger$ resulting in a factor of ~ 4 overhead [717]. Altogether, these considerations suggest that the QSVT-based approach might have a slightly better complexity, particularly when controlled U_H is significantly more costly to implement than U_H. If U_H is already Hermitian, then QSP can have a lower complexity.

Dominant resource cost (gates/qubits)

Using either QSP or QSVT, block-encoding a degree-k polynomial $f(H)$ is performed using $O(k)$ calls to the block-encoding U_H [717, 429]. Hence, the degree of the polynomial approximating e^{iHt} determines the complexity of Hamiltonian simulation using these techniques. As noted in [429, Corollary 60], we can rigorously bound the resources for Hamiltonian simulation via QSVT for all values of t as using

$$O\left(\alpha t + \frac{\log(1/\epsilon)}{\log(e + \log(1/\epsilon)/\alpha t)}\right)$$

calls to the $(\alpha, m, 0)$-block-encoding U_H. This query complexity is optimal [134, 429], although the block-encoding can hide additional complexities, in practice. In some cases, the dependence on norm parameters can be improved by exploiting details of the simulated system; see [715, 714].

For a Pauli access model, the block-encoding is implemented using the linear combination of unitaries (LCU) primitives PREPARE and SELECT. For a Hamiltonian with L terms $\alpha = \|H\|_1$, $m = O(\log(L))$, and two additional qubits are required for QSVT. The overall gate complexity depends on the exact implementation of PREPARE and SELECT, which can often be tailored to the Hamiltonian of interest. In the worst case, PREPARE uses $\Theta(L)$ gates, and SELECT uses $\Theta(nL)$ gates (although these can be significantly improved by exploiting structure in the Hamiltonian; see, e.g., [75, 1011]). Thus, the overall worst-case gate complexity is

$$O\left(nL\left(\|H\|_1 t + \frac{\log(1/\epsilon)}{\log(e + \log(1/\epsilon)/\|H\|_1 t)}\right)\right).$$

For a d-sparse access model, $\alpha = d\|H\|_{\max}$, where $\|H\|_{\max} = \max_{i,j}|\langle i|H|j\rangle|$, $m = O(\log(d))$, and two additional qubits are required for QSVT. The overall gate complexity depends on the cost of sparse access to elements of H. Assuming a circuit for sparse access with constant gate complexity, the overall

gate complexity is

$$O\left(d\|H\|_{\max}t + \frac{\log(1/\epsilon)}{\log(\mathrm{e} + \log(1/\epsilon)/d\|H\|_{\max}t)}\right).$$

Using the QSVT-based amplification technique of [715], the dependence on $d\|H\|_{\max}$ can be improved in some cases.

The density matrix access model seeks to perform time evolution under $\mathrm{e}^{\mathrm{i}\rho t}$, given access to either multiple copies of ρ or a unitary U_ρ that prepares a purification of ρ. Given U_ρ, we can prepare a block-encoding of ρ [717] (see Section 10.1 on block-encodings for details) with $\alpha = 1$. If the gate complexity of U_ρ is $C(U_\rho)$, then the overall gate complexity is

$$O\left(C(U_\rho)\left(t + \frac{\log(1/\epsilon)}{\log(\mathrm{e} + \log(1/\epsilon)/t)}\right)\right).$$

Caveats

The method was found to perform competitively with Trotterization (and better than Taylor series) in concrete resource estimates for simulating spin-chain Hamiltonians [283]. While that work had difficulty calculating the QSP phase factors, this issue has since been addressed with the development of classical algorithms for finding the phase factors (e.g., [431, 477, 356, 255], and successors). Nevertheless, this contributes a classical preprocessing cost to the algorithm.

It is currently unclear how to perform optimal time-dependent Hamiltonian simulation with these methods, without resorting to Trotterization. Some initial investigations have shown promising results using clock Hamiltonian constructions [1027] or for time-periodic Hamiltonians [770, 769].

Example use cases

- Physical systems simulation: quantum chemistry, condensed matter systems (see [283]), quantum field theories, differential equations in plasma physics (see [803]).
- Algorithms: quantum phase estimation, quantum linear system solvers, Gibbs state preparation.

Further reading

- Pedagogical overviews [744, 687].
- Comparison of several Hamiltonian simulation algorithms [283].