Week 5: Hamiltonian Simulation and The HHL algorithm

Contents

1 The Hamiltonian of a system

Before diving into the concept of Hamiltonians, let us recapitulate a couple of points about Quantum state vectors and Unitary matrices.

- Unitary matrices preserve the norm of vectors. This means that as transformations they preserve length and the angle between vectors. You can read more about Unitary Matrices in [1].
- Quantum state vectors $|\psi\rangle$ differ from normal vectors in that they are normalized. This means that their inner product with themselves is unity. $\langle\psi|\psi\rangle = 1$.

The two ways a quantum state can change are either by a **measurement** or by a **unitary transformation**. Measurements are an irreversible process and generally performed as the last step of an experiment, while unitary transformations are reversible operations that occur during an experiment (or a physical process). The unitary transformations which actually occur in a quantum system, or any physical system are determined by the Hamiltonian of the system.
The Hamiltonian $\hat{H}$ is the observable corresponding to the total energy of the system. Hence the Hamiltonian is the sum of the kinetic energies of all the particles, plus the potential energy of the particles associated with the system. Naively we can express it as

$$\begin{aligned} \hat{H} &= \hat{T} + \hat{V} \\ \implies \hat{H} &= -\frac{\hbar}{2m}\nabla^2 + \hat{V} \end{aligned} \tag{1}$$

In reality, the Hamiltonian of an entire system is the sum of many operators which each act on only a part of the system. For example, if our system is a collection of n qubits, $\hat{H}$ can be a sum of up to $\binom{n}{k}$ operators which each act on k qubits at a given time. $\hat{H}$ describes how a quantum state $|\Psi\rangle$ will evolve as a function of time t and the initial state $|\Psi(0)\rangle$.

$$i\hbar\frac{\partial}{\partial t}\Big|\Psi(t)\Big\rangle = \hat{H}\Big|\Psi(t)\Big\rangle \tag{2}$$

The above linear differential equation is known as the Time Dependent Schrödinger Equation. Therefore, if our initial state is $|\Psi(0)\rangle$, the solution to Equation(2) is the following unitary evolution of the state:

$$\Big|\Psi(t)\Big\rangle = exp\left(-\frac{i\hat{H}t}{\hbar}\right)\ \Big|\Psi(0)\Big\rangle \tag{3}$$

where our unitary operator U is

$$U = exp\left(-\frac{iHt}{\hbar}\right) \tag{4}$$

U can also be written[1] as $e^{-i\hat{H}t}$. A few important points to note here:

[1]Here the term $\hbar = h/2\pi$, where h is Planck's constant. For macroscopic systems $\hbar$ is a small number, but for atomic systems it is of order unity. Since we are working with qubits we will be considering $\hbar = 1$ from now on.

- $\hat{H}$ itself maybe time dependent, but for our purposes (designing Hamiltonians for algorithms) we will stick to time independent Hamiltonians.
- Remember that in a physical system t is continuous. However in the quantum computing framework of circuit models we assume t is discrete (and hence an integer).

In his lecture *Simulating Physics with Computers*[2], Feynman envisioned quantum computers as systems that could efficiently simulate every quantum process. In order to simulate a quantum process, we need to efficiently implement the unitary evolution induced by $\hat{H}$. In our context, the t steps of evolution induced by $\hat{H}$, corresponds to applying the unitary $U = e^{-i\hat{H}}$, t times. This, in a nutshell is **Hamiltonian Simulation**.

1.1 Hamiltonian Simulation Techniques

Definition 1.1. H is an n-qubit Hamiltonian if H is a $2^n \times 2^n$ Hermitian matrix.

This shows us that even for reasonable sized systems having 10-15 qubits, the resulting Hamiltonian is pretty big. Suppose we have a Hamiltonian of the form

$$H = \sum_{j=1}^{m} H_j \tag{5}$$

where each H_j only acts on atmost only k of the qubits. This Hamiltonian is called a **k-local Hamiltonian**[2].
Let us consider a 2-local Hamiltonian. Then for a fixed t, each $e^{iH_j t}$ is a 2-qubit gate, that only acts on its chosen qubits and acts as identity on the other $n-2$ qubits. We now wish to implement $U = e^{iHt} = e^{i\sum_j H_j t}$. It is not possible to write this as $U = \prod_j e^{iH_j t}$, since $e^{X+Y} \neq e^X \cdot e^Y$ if X and Y do not commute. The exponential can be computed by the **Lie Product Formula**

$$e^{X+Y} = \lim_{n\to\infty} \left(e^{\frac{X}{n}} e^{\frac{Y}{n}}\right)^n \tag{6}$$

The **Lie-Suzuki-Trotter decomposition** uses this technique to give a unitary for the Hamiltonian form in equation 5.

$$U = e^{iHt} = \left(e^{iHt/r}\right)^r = \left(e^{i\sum_j H_j t/r}\right)^r = \left(\prod_j e^{iH_j t/r} + E\right)^r \tag{7}$$

where $r \geq 1, r \in \mathbb{Z}$, and E is error term with respect to the Lie Product Formula, $\|E\| = O\left(\frac{\prod_j \|H_j\| t^j}{r^j}\right)$. Our circuit approximating U in equation 7 is

$$\tilde{U} = \left(\prod_j e^{iH_j t/r}\right)^r \tag{8}$$

Errors in a product of unitaries add at most linearly. Therefore $\left\|U - \tilde{U}\right\| \leq r^{j-1}\|E\| = O(t^j/r)$. Choosing $r = t^j/\epsilon$, we can get error$\leq \epsilon$. This is the **first order Lie-Suzuki-Trotter** approach to Hamiltonian Simulation. Other Hamiltonian Simulation techniques include **Linear Combination of Unitaries** and **Transforming block-encoded matrices**.

[2]The quantum analogue of the classical SAT problem is the Local Hamiltonian problem. An instance of k-Local Hamiltonian can be viewed as a set of local constraints on n qubits, each involving at most k of them. We are asked to determine if the ground state energy of a given k-local Hamiltonian is less than or greater than the number of expected violated constraints. The 2-Local Hamiltonian problem is QMA-complete, where QMA is the class of all languages that can be probabilistically verified by a quantum verifier in polynomial time. The classical complexity class MA is the randomized analogue of NP. *Modified from* [3]

2 The HHL Algorithm

Classically, we define the **Linear System Problem** as follows

Definition 2.1. Given a matrix[3] $A \in \mathbb{C}^{N\times N}$ and a vector $b \in \mathbb{C}^N$, either output a vector $x \in \mathbb{C}^N$ such that $Ax = b$, or indicate there is no solution..

In the real world, often it suffices to find $\tilde{x}$ which is a "good" approximation of the actual solution x. The HHL algorithm aims to solve the **Quantum Linear System Problem**. In this we take a "well behaved" linear system of equations and output the n-qubit state $|\tilde{x}\rangle$.

Definition 2.2. Find $|\tilde{x}\rangle$ such that $||x\rangle - |\tilde{x}\rangle| \leq \epsilon$, and $A|x\rangle = |b\rangle$.

One obvious drawback of HHL is that we only get back the global state $|\tilde{x}\rangle$. This does not stop the algorithm from being useful, since there are scenarios in which we only need the global state. We will discuss these in some more detail after fleshing out the algorithm. A linear system of equations is called "well behaved" if the following criteria are fulfilled by it:

- A is Hermitian. If A is not Hermitian we convert it into Hermitian form using $A' = \begin{bmatrix} 0 & A \\ A^\dagger & 0 \end{bmatrix}$, where 0 is a 0 matrix of size $N \times N$. Then we can compute $A|\vec{x}\rangle = |\vec{b}\rangle$, where $|\vec{b}\rangle = \begin{bmatrix} |b\rangle \\ 0 \end{bmatrix}$, and $|\vec{x}\rangle = \begin{bmatrix} 0 \\ |x\rangle \end{bmatrix}$.
- A is s-sparse[4] and we have sparse access to elements of A.
- The matrix A is well conditioned. This means that all eigenvalues of A lie in the interval $[-1, -1/\kappa] \cup [1/\kappa, 1]$, where $\kappa = \sigma_{max}/\sigma_{min}$[5]. We assume that our algorithm has an upper bound on κ.
- The state vector $|b\rangle$ is provided to us from another subroutine, or we have a unitary that can efficiently produce this state.

In the classical case, **LSP**'s can be solved using conjugate gradient descent in time $O(Ns\kappa \log 1/\epsilon)$. Here we allow an error of ϵ in the approximation and assume A is s-sparse. HHL solves **QLSP**'s in time $O((\log N)s^2\kappa^2/\epsilon)$. Thus we see that there is a significant tradeoff between exponential speedup in terms of input size, and polynomial slowdown in terms of condition number and sparsity and an exponential slowdown in terms of precision.
We consider A has eigenvectors $|u_j\rangle$ with corresponding eigenvalues λ_j. Hence

- $A = \sum_j \lambda_j u_j u_j^T$.
- For $A|x\rangle = |b\rangle$, the state $|b\rangle$ can be written in terms of eigenvectors of A as $|b\rangle = \sum_j \beta_j |u_j\rangle$.

[3] Assume $N = 2^n$

[4] A has at most s non-zero elements in each row.

[5] κ is the condition number of A and σ refers to the singular values of A. The base-b logarithm of κ is an estimate of how many base-b digits are lost in solving a linear system with that matrix. In other words, it estimates worst-case loss of precision. A system is said to be singular if the condition number is infinite, and ill-conditioned if it is too large.

- $|x\rangle = A^{-1}|b\rangle = \sum_j \beta_j \frac{1}{\lambda_j}|u_j\rangle$. Note that since we have no guarantee that A is unitary, we cannot directly use A^{-1} as a unitary transform in the quantum algorithm.
- Since A is Hermitian, what we do instead is create a unitary $U = e^{iA} = \sum_j e^{i\lambda_j} u_j u_j^T$ via Hamiltonian Simulation techniques, and use this as a unitary transformation in our algorithm.

A rough sketch of the HHL algorithm[4] is given in Algorithm(1).

Algorithm 1: HHL Algorithm

Input : A, $|b\rangle$, t_0, T, ϵ
output: $|x\rangle$

1. Start with two registers C(clock register) and I(input register). The clock register will be responsible for carrying out the Hamiltonian Simulation.

2. Prepare input state $|\Psi_0\rangle^C \otimes |b\rangle^I$. We have

$$|\Psi_0\rangle^C = \sqrt{\frac{2}{T}} \sum_{\tau=0}^{T-1} \sin \frac{\pi(\tau+0.5)}{T} |\tau\rangle \quad \text{, and} \quad |b\rangle^I = \sum_{j=1}^{N} \beta_j |u_j\rangle$$

   Here T refers to the number of simulation steps for simulating e^{iAt}, where $0 \leq t_0 \leq t$. Hence t_0/T is the step size of the simulation. Note that $t_0 = O(\kappa/\epsilon)^a$ and $T = O((\log N)s^2 t_0)$.

3. Now we apply a conditional[b] Hamiltonian $\hat{H}$ on $|\Psi_0\rangle^C \otimes |b\rangle^I$.

$$\hat{H} = \left(\sum_{\tau=0}^{T-1} |\tau\rangle\langle\tau|^C \right) \otimes e^{iA\tau t_0/T}$$

   After rearranging the terms we get,

$$|\Psi_0\rangle^C \otimes |b\rangle^I \overset{\hat{H}}{\mapsto} \sqrt{\frac{2}{T}} \sum_{j=1}^{N} \beta_j \left(\sum_{\tau=0}^{T-1} e^{i\lambda_j \tau t_0/T} \sin \frac{\pi(\tau+0.5)}{T} |\tau\rangle^C \right) |u_j\rangle^I$$

4. We now express the state of the clock register in the Fourier basis $|k\rangle^c$ to get

$$\overset{\text{QFT}}{\longmapsto} \sum_{j=1}^{N} \beta_j \sum_{k=0}^{T-1} \left(\frac{\sqrt{2}}{T} \sum_{\tau=0}^{T-1} e^{i\tau/T(\lambda_j t_0 - -2\pi k)} \sin \frac{\pi(\tau+0.5)}{T} \right) |k\rangle^C |u_j\rangle^I = \sum_{j=1}^{N} \beta_j \left(\sum_{k=0}^{T-1} \alpha_{k|j} \right) |k\rangle^C |u_j\rangle^I$$

   We can show that $\alpha_{k|j}$ is large if and only if $\lambda_j \approx 2\pi k/t_0$. Since only these eigenvalues are of interest[d], let us relabel the basis states $|k\rangle$ by defining $\hat{\lambda}_k = 2\pi k/t_0$.

5. An ancilla qubit S is added to our current state, and then rotated conditioned on our clock register to obtain

$$\sum_{j=1}^{N} \beta_j \left(\sum_{k=0}^{T-1} \alpha_{k|j} \right) |\hat{\lambda}_k\rangle^C |u_j\rangle^I \left(\sqrt{1 - \frac{C^2}{\hat{\lambda}_k^2}} |0\rangle + \frac{C}{\hat{\lambda}_k} |1\rangle \right)^S$$

   C is chosen to be $O(1/\kappa)$. We now undo the phase estimation to uncompute the $|\hat{\lambda}_k\rangle^C$ and assume the phase estimation to be perfect[e].

6. Conditioned on seeing $|1\rangle$, we measure the last register to get $c \sum_{j=1}^{N} \beta_j \frac{C}{\lambda_j} |u_j\rangle$, where c is the normalizing constant. This state corresponds to $|x\rangle = \sum_{j=1}^{N} \beta_j \lambda_j^{-1} |u_j\rangle$.

[a] The calculation for the term t_0 is explained later on in VTAA.

[b] $\hat{H}$ is conditional since the length of the simulation is conditioned on the value of the clock register.

[c] by taking an inner product with $\frac{1}{\sqrt{T}} \sum_{\tau=0}^{T-1} e^{-2\pi i k\tau/T} |k\rangle$

The runtime of HHL can be calculated as follows

- Each run of HHL concludes in T steps.
- The success probability of the post-selection process is atleast $O(1/\kappa^2)$ since C is chosen to be $O(1/\kappa)$. Hence to get a good success probability we need to perform $O(\kappa)$ amplifications[6].

Putting both of these points together we see that the runtime of HHL is

$$O(\kappa T) = O(\kappa(\log N)s^2 t_0) = O\left(\frac{(\log N)s^2\kappa^2}{\epsilon}\right)$$

HHL algorithm has its pros and cons. Classically solving linear equations meant getting the values $x_1, ..., x_N$ from the quantum state $|x\rangle$. In the quantum case, even estimating these values would mean producing many copies of $|x\rangle$, thereby increasing the running time significantly. On the other hand, the global state $|x\rangle$ can be used to estimate expressions which depend on all x_i simultaneously. Classically, it is not clear how we would go about estimating expressions of this form without solving for the individual terms $x_1, ..., x_N$. The global state $|x\rangle$ is also handy when we want to test how similar the solutions of two systems of linear equations $Ax = b$ and $A'x' = b'$ are. Using HHL we simply generate $|x\rangle$ and $|x'\rangle$ and then compare them using the SWAP-test.

2.1 Variable Time Amplitude Amplification

Andris Ambainis came up with an improvement[5] to the quadratic dependence on the condition number κ in the HHL algorithm. The new running time is

$$O\left(\frac{(\log N)s^2\kappa\log^3\kappa}{\epsilon}\right)$$

The term κ^2 is generated as a product of two κ's.

- For $||\psi - \psi'|| \leq \epsilon$, it suffices that the estimates of the eigenvalues, $\hat{\lambda}_i$ satisfy $|\lambda_i - \hat{\lambda}_i| = O(\epsilon\hat{\lambda}_i)$. Since for "well defined" linear system of equations we have $\lambda_i = \Omega\left(\frac{1}{\kappa}\right)$, it implies that $|\lambda_i - \hat{\lambda}_i| = O\left(\frac{\epsilon}{\kappa}\right)$. Therefore to estimate λ_i within error $O\left(\frac{\epsilon}{\kappa}\right)$, we need to run the Hamiltonian $O\left(\frac{\kappa}{\epsilon}\right)$ times. This is where our t_0 term comes in.
- For amplitude amplification we may need to repeat the algorithm $O(\kappa)$ times.

To understand the intuition behind the VTAA algorithm we need to analyse these two conditions in a little bit more depth

- The worst case of eigenvalue estimation occurs when most λ_i are very small, i.e. $\lambda_i = \Theta\left(\frac{1}{\kappa}\right)$. In this case we need to run the Hamiltonian $\Theta\left(\frac{\kappa}{\epsilon}\right)$ times.

[6]Grover's amplitude amplification

- The worst case of amplitude amplification occurs when most λ_i are large (near 1). From step 5 of Algorithm 1, we see that the coefficient corresponding to the good state is

$$\frac{C}{\hat{\lambda}_i} = \frac{1}{\kappa\hat{\lambda}_i} = \Theta\left(\frac{1}{\kappa}\right)$$

  Hence we need $\Theta(\kappa)$ rotations for amplifying the probability of the good state.

From the above analysis, we can see that the two terms of κ occur in completely contradicting scenarios. If our eigenvalues have similar magnitudes (i.e. belong to a constant range $[a, ca]$, for some a and constant c), we can show that the running time is

$$O\left(\frac{1}{a\epsilon}\right) \times O(a\kappa) = O\left(\frac{\kappa}{\epsilon}\right)$$

as the eigenvalue estimation has error $a\epsilon$ and, the amplitude of 1 is $\Theta\left(\frac{1}{a\kappa}\right)$ in step 5 of Algorithm 1. Since we have established that the running time of HHL is not set in stone

by using a constrained range of eigenvalues, the next task ahead is to generalize it for all $\lambda_i \in \left[\frac{1}{\kappa}, 1\right]$. We encourage the reader to peruse the original paper by A. Ambainis[5] to explore this train of thought further. We will be exploring a bit more of this topic in the exercises.

3 Exercises

1. We will try proving some basic properties about Unitary Matrices.
   (a) Prove that unitary operations preserve the norm of a vector.
   (b) Prove that unitary operations preserve the inner product between two vectors.
2. Compute the unitaries: e^{iX}, $e^{i\sigma_x}e^{i\sigma_z}$ and $e^{i(\sigma_x+\sigma_z)}$, where σ_x and σ_z are the Pauli-X and Pauli-Z matrices respectively. Show that $e^{i\sigma_x}e^{i\sigma_z}$ and $e^{i(\sigma_x+\sigma_z)}$ are not equal (as discussed previously).
3. If A is skew-Hermitian, prove that e^{iA} is unitary.
4. Suppose we want to implement a certain unitary U, and we can do that by switching on a Hamiltonian H for some time t: $U = e^{-iHt}$. Now suppose $\tilde{H}$ is another Hamiltonian, with 100 times as much energy as H: $\tilde{H} = 100H$. Show that using $\tilde{H}$ we can implement U a 100 times faster than with H[7].
5. In Algorithm1, write the state obtained after step 5 (i.e., after uncomputing the clock register and assuming perfect phase estimation).
6. In the section on HHL, we discussed comparing two quantum states using the SWAP-test (refer Figure 1).

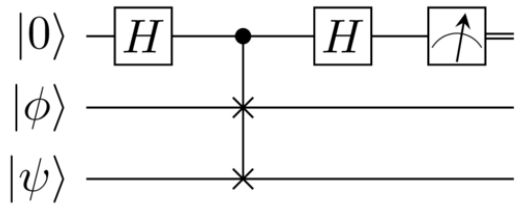

Figure 1: The SWAP test using a Fredkin's gate.[8]

   (a) Suppose we apply two identical qubits to the second and third registers. What is the probability of obtaining $|0\rangle$?
   (b) Suppose we apply two orthogonal qubits to the second and third registers. What is the probability of obtaining $|0\rangle$?

[7]Taken from Ronald de Wolf's lecture notes(Chapter 9, Exercise 2).
[8]Image Credit: Vtomole [CC BY-SA 4.0(https://creativecommons.org/licenses/by-sa/4.0)]

7. Design an algorithm where we selectively estimate the estimates $\hat{\lambda}_i$ for each eigenvalue λ_i.

8. Suppose all λ_i lie in the range $[a^c, a]$. What is the minimum value of the constant c such that a relatively tighter bound gives us a better running time than HHL? (Assume $a \leq 1 - o(1)$).

References

[1] G. D. Allen, "Math 640, linear algebra for applications, lecture 4," Fall 2003. [Online]. Available: https://www.math.tamu.edu/~dallen/m640_03c/lectures/chapter4.pdf

[2] R. P. Feynman, "Simulating physics with computers," *Int. J. Theor. Phys.*, vol. 21, pp. 467–488, 1982, [,923(1981)].

[3] J. Kempe, A. Kitaev, and O. Regev, "The complexity of the local hamiltonian problem," 2004.

[4] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for linear systems of equations," *Phys. Rev. Lett.*, vol. 103, p. 150502, Oct 2009. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.103.150502

[5] A. Ambainis, "Variable time amplitude amplification and a faster quantum algorithm for solving systems of linear equations," 2010.