PHYC90045 Introduction to Quantum Computing

Week 11

Lecture 21
Adiabatic Quantum Computation

Lecture 22
Further quantum algorithms – HHL algorithm

Lab 11
Adiabatic Quantum Computation

1

PHYC90045 Introduction to Quantum Computing

Solving Linear Equations

Physics 90045
Lecture 22

2

PHYC90045 Introduction to Quantum Computing

Overview

This lecture we will introduce our final quantum computing algorithm for the course:

- Solving Linear Equations
- Mapping Linear Equations to a Quantum Computer
- Solving them: The HHL Algorithm

Introduction paper:
https://arxiv.org/pdf/1802.08227.pdf

3

Solving Linear Equations is Ubiquitous!

So many areas of application:

- Finance
- Engineering (eg. Electronics, civil engineering)
- Defence (eg. Radar)
- Science (Least squares optimization, ODE solving…)
- Machine learning, control theory
- … so many more

4

PHYC90045 Introduction to Quantum Computing

Linear Equations

$$\boxed{\begin{aligned} 2x + y &= 5 \\ x - 2y &= 0 \end{aligned}}$$

What is the solution for x and y?

5

PHYC90045 Introduction to Quantum Computing

Solving Linear Equations

Write as a matrix equation:

$$\boxed{Ax = b}$$

In our case:

$$\boxed{\begin{bmatrix} 2 & 1 \\ 1 & -2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 5 \\ 0 \end{bmatrix}}$$

Can solve by row reduction, or by finding the inverse of A.
We will find the inverse of A.

6

PHYC90045 Introduction to Quantum Computing

Calculating Matrix Inverse Using Eigenvalues

We would like to invert the matrix A:

$$A = \begin{bmatrix} 2 & 1 \\ 1 & -2 \end{bmatrix}$$

To find the eigenvalues, we will first write the characteristic equation:

$$|A - \lambda I| = \begin{vmatrix} 2-\lambda & 1 \\ 1 & -2-\lambda \end{vmatrix} = \lambda^2 - 5$$

And find the roots:

$$\lambda^2 - 5 = 0$$
$$\therefore \lambda = \pm\sqrt{5}$$

7

PHYC90045 Introduction to Quantum Computing

Matrix and Matrix Inverse

$$A = VDV^\dagger$$

Change back

Change to eigenbasis

$$D = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \qquad D^{-1} = \begin{bmatrix} \frac{1}{\lambda_1} & 0 \\ 0 & \frac{1}{\lambda_2} \end{bmatrix}$$

Inverse of A is $A^{-1} = VD^{-1}V^\dagger$

Checking:

$$\begin{aligned} AA^{-1} &= VDV^\dagger VD^{-1}V^\dagger \\ &= VDD^{-1}V^\dagger \\ &= VV^\dagger \\ &= I \end{aligned}$$

8

PHYC90045 Introduction to Quantum Computing

Eigenvalues to Inverse

We can find the corresponding eigenvectors (exercise for you!):

$$\lambda_1 = -\sqrt{5}, \quad u_1 = \begin{bmatrix} 2-\sqrt{5} \\ 1 \end{bmatrix}$$

$$\lambda_2 = \sqrt{5}, \quad u_2 = \begin{bmatrix} 2+\sqrt{5} \\ 1 \end{bmatrix}$$

Based on the previous slide we can find the inverse:

$$A^{-1} = VD^{-1}V^\dagger$$

$$A^{-1} = \frac{1}{5}\begin{bmatrix} 2 & 1 \\ 1 & -2 \end{bmatrix}$$

9

PHYC90045 Introduction to Quantum Computing

Invert the matrix

$$Ax = b$$
$$A^{-1}Ax = A^{-1}b$$
$$x = A^{-1}b$$

So in our case:

$$x = \frac{1}{5}\begin{bmatrix} 2 & 1 \\ 1 & -2 \end{bmatrix}\begin{bmatrix} 5 \\ 0 \end{bmatrix}$$
$$= \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

And we've solved the linear equations. The HHL quantum algorithm works in a similar way on a quantum computer.

10

PHYC90045 Introduction to Quantum Computing

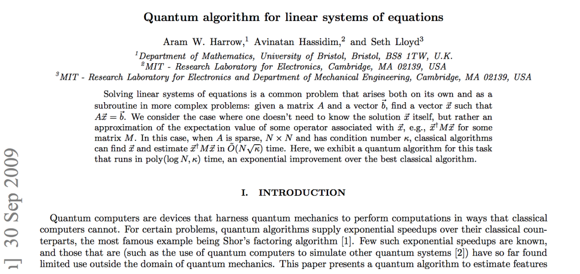
Steps for solving classically

1) Write as a matrix
2) Find eigenvalues of the matrix, A
3) Use eigenvalues to invert matrix, A^{-1}
4) Apply A^{-1} to b

11

PHYC90045 Introduction to Quantum Computing

HHL Algorithm


Quantum algorithm for linear systems of equations

Aram W. Harrow,[1] Avinatan Hassidim,[2] and Seth Lloyd[3]
[1]Department of Mathematics, University of Bristol, Bristol, BS8 1TW, U.K.
[2]MIT - Research Laboratory for Electronics, Cambridge, MA 02139, USA
[3]MIT - Research Laboratory for Electronics and Department of Mechanical Engineering, Cambridge, MA 02139, USA

Solving linear systems of equations is a common problem that arises both on its own and as a subroutine in more complex problems: given a matrix A and a vector $\vec{b}$, find a vector $\vec{x}$ such that $A\vec{x} = \vec{b}$. We consider the case where one doesn't need to know the solution $\vec{x}$ itself, but rather an approximation of the expectation value of some operator associated with $\vec{x}$, e.g., $\vec{x}^\dagger M\vec{x}$ for some matrix M. In this case, when A is sparse, $N \times N$ and has condition number κ, classical algorithms can find $\vec{x}$ and estimate $\vec{x}^\dagger M\vec{x}$ in $\tilde{O}(N\sqrt{\kappa})$ time. Here, we exhibit a quantum algorithm for this task that runs in poly$(\log N, \kappa)$ time, an exponential improvement over the best classical algorithm.

I. INTRODUCTION

Quantum computers are devices that harness quantum mechanics to perform computations in ways that classical computers cannot. For certain problems, quantum algorithms supply exponential speedups over their classical counterparts, the most famous example being Shor's factoring algorithm [1]. Few such exponential speedups are known, and those that are (such as the use of quantum computers to simulate other quantum systems [2]) have so far found limited use outside the domain of quantum mechanics. This paper presents a quantum algorithm to estimate features

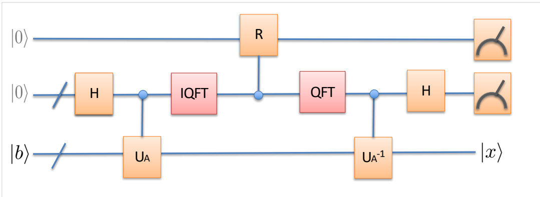
1] 30 Sep 2009

Phys. Rev. Lett. vol. 15, no. 103, pp. 150502 (2009)
arxiv.org 0811.3171

12

PHYC90045 Introduction to Quantum Computing

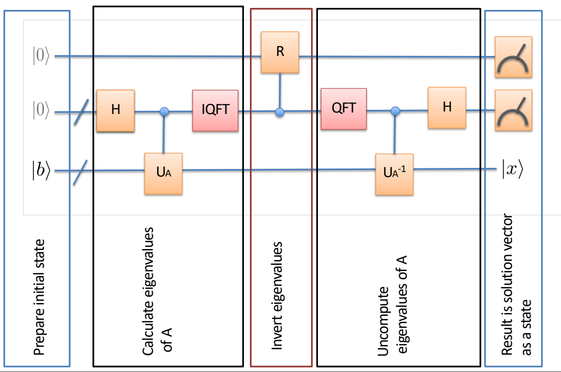
HHL Algorithm



13

PHYC90045 Introduction to Quantum Computing

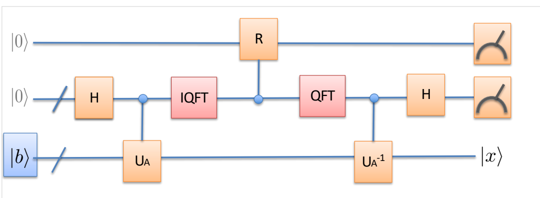
Structure of the HHL Algorithm



14

PHYC90045 Introduction to Quantum Computing

Initial State



Set up a quantum state represented by the normalized vector "b". In our example:

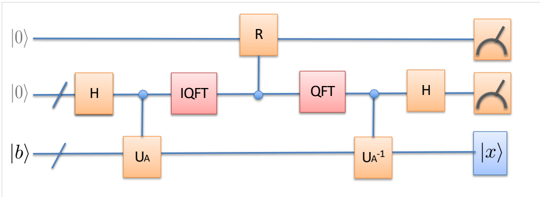
$$b = \begin{bmatrix} 5 \\ 0 \end{bmatrix} \xrightarrow{\text{Normalize}} |b\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle$$

Input might be a superposition, and possibly difficult to prepare!

15

PHYC90045 Introduction to Quantum Computing

Final State



Output is a quantum state representing the normalized vector "x". In our example:

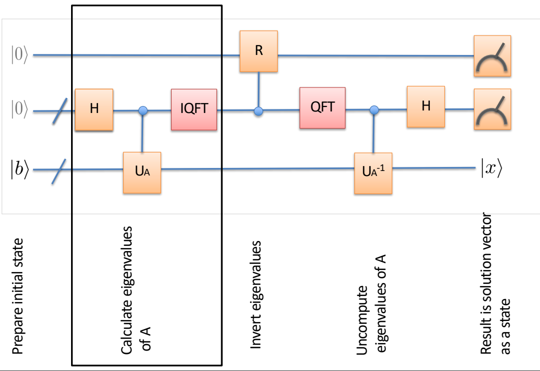
$$|x\rangle = \frac{1}{\sqrt{5}}\begin{bmatrix} 2 \\ 1 \end{bmatrix} \xrightarrow{\text{Corresponds to}} x = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Warning: Output is as a **superposition**.

16

PHYC90045 Introduction to Quantum Computing

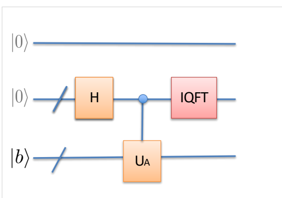
Calculate the Eigenvalues of A



17

PHYC90045 Introduction to Quantum Computing

Calculate the Eigenvalues of A



Looks a lot like Shor's algorithm/Simon's algorithm which we have seen previously:

- Initial equal superposition
- Application of some function, in this case U_A
- Inverse QFT

18

U_A

$$U_A = \exp(iAt)$$

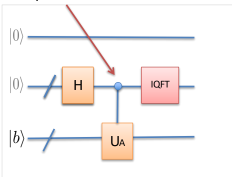
How to exponentiate a general A as a circuit? One way if A is Hermitian:

- Break up as Pauli matrices
- Use Trotter!

In our case:

$$A = \begin{bmatrix} 2 & 1 \\ 1 & -2 \end{bmatrix}$$
$$= X + 2Z$$

t is the state of the control qubit!



19

Matrix as linear combination of Pauli operators

Always possible decompose a matrix as a sum of Paulis. If you have a matrix only:

$$E_i = \frac{\mathrm{Tr}\left[\sigma_i H\right]}{d}$$

Where d is the dimension of the system, H is the Hamiltonian and σ_i is the Pauli. If the matrix is Hermitian, the co-efficients you find, E_i, should be real.

Express the Hamiltonian as linear combination of Pauli matrices:

$$H = \sum_i E_i \sigma_i$$

For example: $H = B_1 X_1 + B_2 X_2 + J_{12} Z_1 Z_2$

20

Trotter Approximation

But what if you do want to create the gate:

$$\exp(A+B)$$

We might try:

$$\exp(A+B) \approx \exp(A)\exp(B)$$

$$\exp(A+B) \approx \exp\left(\frac{A}{2}\right)\exp\left(\frac{B}{2}\right)\exp\left(\frac{A}{2}\right)\exp\left(\frac{B}{2}\right)$$

$$\exp(A+B) \approx \left(\exp\left(\frac{A}{n}\right)\exp\left(\frac{B}{n}\right)\right)^n$$

This is called the Trotter (sometimes Trotter-Suzuki) approximation – useful!

21

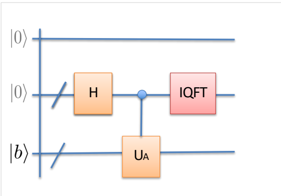
If A is not Hermitian

If A is not Hermitian, can make it Hermitian:

$$A' = \begin{bmatrix} 0 & A \\ A^\dagger & 0 \end{bmatrix}$$

22

Step by Step

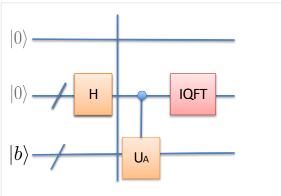


Initially, the state is:

$$|\psi\rangle = |0\rangle\,|0\rangle\,|b\rangle$$

23

Step by Step

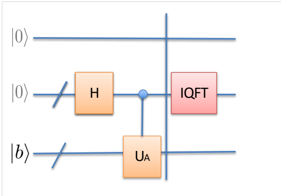


After the Hadamard gates the "t" register is in an equal superposition:

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{t=0}^{N-1} |0\rangle\,|t\rangle\,|b\rangle$$

24

Step by Step



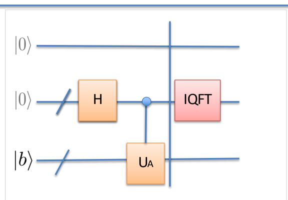
We then apply the U_A gate:

$$|\psi\rangle = \frac{1}{\sqrt{N}}\sum_{t=0}^{N-1}|0\rangle\,|t\rangle\;\exp(iAt)\,|b\rangle$$

Angle depends on the "t" register.

25

Step by Step



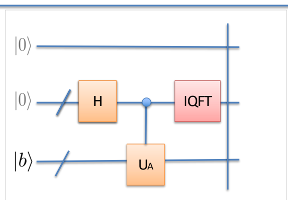
For now let us imagine that b is an eigenstate of a (we will remove this assumption later)

$$|\psi\rangle = \frac{1}{\sqrt{N}}\sum_{t=0}^{N-1}|0\rangle\,|t\rangle\;\exp(iAt)\,|b\rangle$$
$$= \frac{1}{\sqrt{N}}\sum_{t=0}^{N-1}|0\rangle\;\exp(i\lambda_b t)\,|t\rangle\,|b\rangle$$

Phase kickback on the the "t" register. Periodic according to the eigenvalue!

26

Step by Step



Applying IQFT extracts the eigenvalue:

$$|\psi\rangle = U_{IQFT}\frac{1}{\sqrt{N}}\sum_{t=0}^{N-1}|0\rangle\;\exp(i\lambda_b t)\,|t\rangle\,|b\rangle$$
$$= |0\rangle\,|\tilde{\lambda}_b\rangle\,|b\rangle$$

Giving an approximation to the Eigenvalue in the "t" register.

27

Superposition of Eigenstates

If, instead of a single Eigenstate, b was made up of a superposition of Eigenstates of A:

$$|b\rangle = \sum_i b_i |u_i\rangle$$

Then so too, after performing the algorithm, would we be in a superposition of states:

$$|\psi\rangle = \sum_i b_i |0\rangle |\tilde{\lambda}_i\rangle |u_i\rangle$$

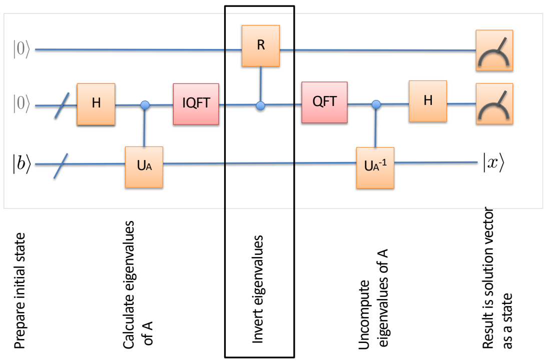
What we want is the solution, which depends on quantities we have calculated:

$$|x\rangle = \sum_i \frac{b_i}{\lambda_i} |u_i\rangle$$

How can we reduce each amplitude b_i by a factor of *λ?* This is *not even unitary!*

28

Calculate the Eigenvalues of A



29

Inverting the Eigenvalues

The t register contains register contains the eigenvalue. Based on this register we can make a controlled rotation on the ancilla qubit to obtain the state (of the ancilla register):

$$\sqrt{1-\frac{C^2}{\tilde{\lambda}^2}}|0\rangle + \frac{C}{\tilde{\lambda}}|1\rangle$$

R_y by

$$\theta = -2\cos^{-1}\left(\frac{C}{\tilde{\lambda}}\right)$$

We now measure. If we obtain the state 0, we redo the algorithm ("post-select") until we have measured the 1 state.

Initially the state is: $|\psi\rangle = \sum_i b_i |0\rangle |\tilde{\lambda}_i\rangle |u_i\rangle$

Becomes:

$$\sum_i b_i \left(\sqrt{1-\frac{C^2}{\tilde{\lambda}^2}}|0\rangle + \frac{C}{\tilde{\lambda}}|1\rangle\right)|\tilde{\lambda}_i\rangle |u_i\rangle$$

30

Applied to our state

$$|\psi\rangle = \sum_i b_i |0\rangle |\tilde{\lambda}_i\rangle |u_i\rangle$$

After rotation, R:

$$|\psi\rangle = \sum_i b_i \left(\sqrt{1 - \frac{C^2}{\tilde{\lambda}^2}} |0\rangle + \frac{C}{\tilde{\lambda}} |1\rangle \right) |\tilde{\lambda}_i\rangle |u_i\rangle$$

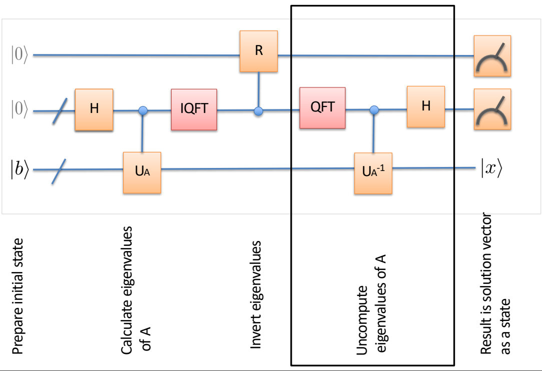
And after post-selecting based on measuring the state 1,

Badly normalized: $$|\psi\rangle = \sum_i b_i \frac{C}{\tilde{\lambda}} |1\rangle |\tilde{\lambda}_i\rangle |u_i\rangle$$

$$|\psi\rangle = \sum_i \frac{b_i}{\tilde{\lambda}} |1\rangle |\tilde{\lambda}_i\rangle |u_i\rangle$$

31

"Uncompute" the Eigenvalues



32

Invert the eigenvalues

From our state:

$$|\psi\rangle = \sum_i \frac{b_i}{\tilde{\lambda}} |1\rangle |\tilde{\lambda}_i\rangle |u_i\rangle$$

We apply the opposite procedure which we used to calculate the eigenvalues.
This resets them to the zero state:

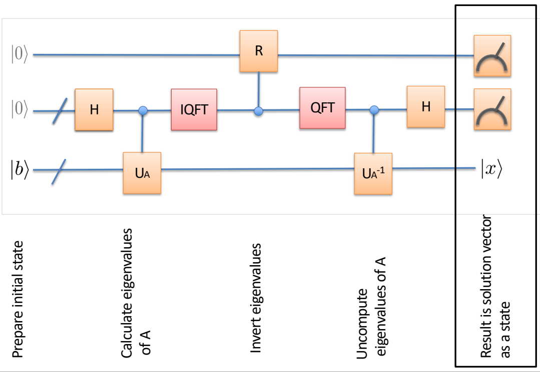
$$|\psi\rangle = \sum_i \frac{b_i}{\tilde{\lambda}} |1\rangle |0\rangle |u_i\rangle$$

Compare to the solution:

$$|x\rangle = \sum_i \frac{b_i}{\lambda_i} |u_i\rangle$$

33

"Uncompute" the Eigenvalues



34

Obtain solution

Obtain an approximation to the solution:

$$|\tilde{x}\rangle = \sum_i \frac{b_i}{\tilde{\lambda}} |u_i\rangle$$

Notice this is as a superposition, so we are left with a state whose vector represents the answer to the problem. Completely determining this state, in itself can be a costly exercise, **potentially requiring an exponentially large number of measurements**. However it can be used in other ways, such as for calculating average energy, or as an input to another such function.

35

Condition Number

The condition number, κ, of a matrix A is given by the ratio of largest to smallest eigenvalues,

$$\kappa = \lambda_{max}/\lambda_{min}$$

36

Running time

Best general classical algorithm is the conjugate gradient method, with runtime:

$$O\left(Ns\kappa\log(1/\epsilon)\right)$$

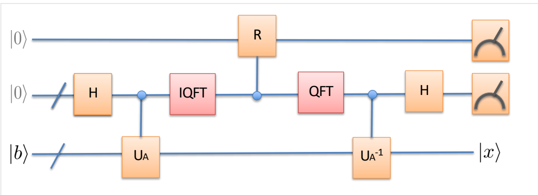
HHL algorithm has runtime:

$$O\left(\log(N)s^2\kappa^2/\epsilon\right)$$

Where N is the dimension of the matrix
s is the sparsity, κ is the condition number of A, and ϵ is the error

37

Summary



- HHL algorithm solves systems of linear equations
- Need a method for emulating *exp(iAt)*
- Can be used as a quantum subroutine
- Has a runtime, exponentially faster than classical versions.

38

Week 11

Lecture 21
Adiabatic Quantum Computation

Lecture 22
- Further quantum algorithms – HHL algorithm

Lab 11
Adiabatic Quantum Computation

39

