量子計算的數學基礎 MA5501*

Chapter 8. The HHL Algorithm

§8.1 The Linear System Problem

In this chapter we present the **Harrow-Hassidim-Lloyd (HHL) algorithm** for solving large systems of linear equations. Such a system is given by an $N \times N$ matrix A with real or complex entries, and an N-dimensional nonzero vector b. Assume for simplicity that $N = 2^n$. The linear-system problem is

LSP: find an N-dimensional vector x such that $Ax = b$.

Solving large systems of linear equations is extremely important in many computational problems in industry, in science, in optimization, in machine learning, etc. In many applications it suffices to find a vector $\tilde{x}$ that is close to the actual solution x.

§8.1 The Linear System Problem

We will assume A is invertible (equivalently, has rank N) in order to guarantee the existence of a unique solution vector x, which is then just $A^{-1}b$. This assumption is just for simplicity: if A does not have full rank, then the methods below would still allow to invert it on its support, replacing A^{-1} by the "Moore-Penrose pseudoinverse".

The HHL algorithm can solve "well-behaved" large linear systems very fast (under certain assumptions), but in a rather weak sense: instead of outputting the N-dimensional solution vector x itself, its goal is to output the n-qubit state

$$|x\rangle = \frac{1}{\|x\|} \sum_{i=0}^{N-1} x_i |i\rangle$$

or some other n-qubit state close to $|x\rangle$.

§8.1 The Linear System Problem

We will assume A is invertible (equivalently, has rank N) in order to guarantee the existence of a unique solution vector x, which is then just $A^{-1}b$. This assumption is just for simplicity: if A does not have full rank, then the methods below would still allow to invert it on its support, replacing A^{-1} by the "Moore-Penrose pseudoinverse".

The HHL algorithm can solve "well-behaved" large linear systems very fast (under certain assumptions), but in a rather weak sense: instead of outputting the N-dimensional solution vector x itself, its goal is to output the n-qubit state

$$|x\rangle = \frac{1}{\|x\|}\sum_{i=0}^{N-1} x_i|i\rangle$$

or some other n-qubit state close to $|x\rangle$.

§8.1 The Linear System Problem

This state $|x\rangle$ has the solution vector as its vector of amplitudes, up to normalization. This is called the quantum linear-system problem:

QLSP: find an n-qubit state $|\widetilde{x}\rangle$ such that $\||x\rangle - |\widetilde{x}\rangle\| \leqslant \varepsilon$ and $Ax = b$.

Note that the QLSP is an inherently quantum problem, since the goal is to produce an n-qubit state whose amplitude-vector (up to normalization and up to ε-error) is a solution to the linear system. In general this is not as useful as just having the N-dimensional vector x written out on a piece of paper, but in some cases where we only want some partial information about x, it may suffice to just (approximately) construct $|x\rangle$.

§8.1 The Linear System Problem

We will assume without loss of generality that A is Hermitian: if A is a non-hermitian $N \times N$ matrix, then we consider the augmented linear system (of size $2N$) $\bar{A}\bar{x} = \bar{b}$, where with $\mathbf{0}_{N\times N}$ denoting the $N \times N$ zero matrix and $\mathbf{0}_{N\times 1}$ denoting the zero (column) vector in $\mathbb{R}^N$,

$$\bar{A} \equiv \begin{bmatrix} \mathbf{0}_{N\times N} & A \\ A^{\dagger} & \mathbf{0}_{N\times N} \end{bmatrix}, \quad \bar{b} = \begin{bmatrix} b \\ \mathbf{0}_{N\times 1} \end{bmatrix}.$$

Note that if x solves $Ax = b$ (or equivalently, $x = A^{-1}b$), then $\bar{x}$ takes the form $\bar{x} = \begin{bmatrix} \mathbf{0}_{N\times 1} \\ x \end{bmatrix}$.

§8.1 The Linear System Problem

We will assume without loss of generality that A is Hermitian: if A is a non-hermitian $N \times N$ matrix, then we consider the augmented linear system (of size $2N$) $\bar{A}\bar{x} = \bar{b}$, where with $\mathbf{0}_{N\times N}$ denoting the $N \times N$ zero matrix and $\mathbf{0}_{N\times 1}$ denoting the zero (column) vector in $\mathbb{R}^N$,

$$\bar{A} \equiv \begin{bmatrix} \mathbf{0}_{N\times N} & A \\ A^\dagger & \mathbf{0}_{N\times N} \end{bmatrix}, \quad \bar{b} = \begin{bmatrix} b \\ \mathbf{0}_{N\times 1} \end{bmatrix}.$$

Note that if x solves $Ax = b$ (or equivalently, $x = A^{-1}b$), then $\bar{x}$ takes the form $\bar{x} = \begin{bmatrix} \mathbf{0}_{N\times 1} \\ x \end{bmatrix}$.

§8.1 The Linear System Problem

Let us state the more restrictive assumptions that will make the linear system "well-behaved" and suitable for the HHL algorithm:

1. We have a unitary that can prepare the vector b as an n-qubit quantum state
$$|b\rangle = \frac{1}{\|b\|}\sum_{i=0}^{N-1} b_i|i\rangle$$
using a circuit of B 2-qubit gates. We also assume for simplicity that $\|b\| = 1$.
2. The matrix A is s-sparse and we have sparse access to it. Such sparsity is not essential to the algorithm, and could be replaced by other properties that enable an efficient block-encoding of A.

§8.1 The Linear System Problem

Let us state the more restrictive assumptions that will make the linear system "well-behaved" and suitable for the HHL algorithm:

1. We have a unitary that can prepare the vector b as an n-qubit quantum state
$$|b\rangle = \frac{1}{\|b\|}\sum_{i=0}^{N-1} b_i|i\rangle$$
using a circuit of B 2-qubit gates. We also assume for simplicity that $\|b\| = 1$.
2. The matrix A is s-sparse and we have sparse access to it. Such sparsity is not essential to the algorithm, and could be replaced by other properties that enable an efficient block-encoding of A.

§8.1 The Linear System Problem

3. The matrix A is well-conditioned: the ratio between its largest and smallest singular value is at most some κ. For simplicity, assume the smallest singular value is not smaller than $1/\kappa$ while the largest is not greater than 1. In other words, all eigenvalues of A lie in the interval $[-1,-1/\kappa] \cup [1/\kappa, 1]$. The smaller the "condition number" κ is, the better it will be for the algorithm. Let us assume our algorithm knows κ, or at least knows a reasonable upper bound on κ.

§8.1 The Linear System Problem

3. The matrix A is well-conditioned: the ratio between its largest and smallest singular value is at most some κ. For simplicity, assume the smallest singular value is not smaller than $1/\kappa$ while the largest is not greater than 1. In other words, all eigenvalues of A lie in the interval $[-1,-1/\kappa] \cup [1/\kappa, 1]$. The smaller the "condition number" κ is, the better it will be for the algorithm. Let us assume our algorithm knows κ, or at least knows a reasonable upper bound on κ.

§8.2 The Basic HHL Algorithm for Linear Systems

Let us start with some intuition. The solution vector x that we are looking for is $A^{-1}b$, so we would like to apply A^{-1} to b. Since A is hermitian, A has spectral decomposition $A = \sum_{j=0}^{N-1} \lambda_j a_j a_j^\dagger$; then the map A^{-1} is the same as the map $a_j \mapsto \frac{1}{\lambda_j} a_j$: we just want to multiply the eigenvector a_j with the scalar $1/\lambda_j$. The vector b can also be written as a linear combination of the eigenvectors a_j: $b = \sum_{j=0}^{N-1} \beta_j a_j$ (we do **NOT** need to know the coefficients β_j for what follows). We want to apply A^{-1} to b to obtain $A^{-1}b = \sum_{j=0}^{N-1} \beta_j \frac{1}{\lambda_j} a_j$, **normalized**, as an n-qubit quantum state.

§8.2 The Basic HHL Algorithm for Linear Systems

Let us start with some intuition. The solution vector x that we are looking for is $A^{-1}b$, so we would like to apply A^{-1} to b. Since A is hermitian, A has spectral decomposition $A = \sum_{j=0}^{N-1} \lambda_j a_j a_j^\dagger$; then the map A^{-1} is the same as the map $a_j \mapsto \frac{1}{\lambda_j} a_j$: we just want to multiply the eigenvector a_j with the scalar $1/\lambda_j$. The vector b can also be written as a linear combination of the eigenvectors a_j: $b = \sum_{j=0}^{N-1} \beta_j a_j$ (we do **NOT** need to know the coefficients β_j for what follows). We want to apply A^{-1} to b to obtain $A^{-1}b = \sum_{j=0}^{N-1} \beta_j \frac{1}{\lambda_j} a_j$, **normalized**, as an n-qubit quantum state.

§8.2 The Basic HHL Algorithm for Linear Systems

Let us start with some intuition. The solution vector x that we are looking for is $A^{-1}b$, so we would like to apply A^{-1} to b. Since A is hermitian, A has spectral decomposition $A = \sum_{j=0}^{N-1} \lambda_j a_j a_j^\dagger$; then the map A^{-1} is the same as the map $a_j \mapsto \frac{1}{\lambda_j} a_j$: we just want to multiply the eigenvector a_j with the scalar $1/\lambda_j$. The vector b can also be written as a linear combination of the eigenvectors a_j: $b = \sum_{j=0}^{N-1} \beta_j a_j$ (we do **NOT** need to know the coefficients β_j for what follows). We want to apply A^{-1} to b to obtain $A^{-1}b = \sum_{j=0}^{N-1} \beta_j \frac{1}{\lambda_j} a_j$, **normalized**, as an n-qubit quantum state.

§8.2 The Basic HHL Algorithm for Linear Systems

Unfortunately the maps A and A^{-1} are not unitary (unless $|\lambda_j| = 1$ for all j), so we cannot just apply A^{-1} as a quantum operation to state $|b\rangle$ to get state $|x\rangle$. Fortunately $U = e^{iA} = \sum_{j=0}^{N-1} e^{i\lambda_j} a_j a_j^\dagger$ is unitary, and has the same eigenvectors as A and A^{-1}. We can implement U and powers of U by Hamiltonian simulation, and then use phase estimation to estimate the λ_j associated with eigenvector $|a_j\rangle$ with some small approximation error.

Conditioned on our estimate of λ_j we can then rotate an auxiliary $|0\rangle$-qubit to

$$\sqrt{1 - \frac{1}{\kappa^2 \lambda_j^2}}|0\rangle + \frac{1}{\kappa\lambda_j}|1\rangle$$

(this is a valid state because $|\kappa\lambda_j| \geqslant 1$).

§8.2 The Basic HHL Algorithm for Linear Systems

Unfortunately the maps A and A^{-1} are not unitary (unless $|\lambda_j| = 1$ for all j), so we cannot just apply A^{-1} as a quantum operation to state $|b\rangle$ to get state $|x\rangle$. Fortunately $U = e^{iA} = \sum_{j=0}^{N-1} e^{i\lambda_j} a_j a_j^\dagger$ is unitary, and has the same eigenvectors as A and A^{-1}. We can implement U and powers of U by Hamiltonian simulation, and then use phase estimation to estimate the λ_j associated with eigenvector $|a_j\rangle$ with some small approximation error.

Conditioned on our estimate of λ_j we can then rotate an auxiliary $|0\rangle$-qubit to

$$\sqrt{1 - \frac{1}{\kappa^2 \lambda_j^2}}|0\rangle + \frac{1}{\kappa\lambda_j}|1\rangle$$

(this is a valid state because $|\kappa\lambda_j| \geqslant 1$).

§8.2 The Basic HHL Algorithm for Linear Systems

Unfortunately the maps A and A^{-1} are not unitary (unless $|\lambda_j| = 1$ for all j), so we cannot just apply A^{-1} as a quantum operation to state $|b\rangle$ to get state $|x\rangle$. Fortunately $U = e^{iA} = \sum_{j=0}^{N-1} e^{i\lambda_j} a_j a_j^\dagger$ is unitary, and has the same eigenvectors as A and A^{-1}. We can implement U and powers of U by Hamiltonian simulation, and then use phase estimation to estimate the λ_j associated with eigenvector $|a_j\rangle$ with some small approximation error.

Conditioned on our estimate of λ_j we can then rotate an auxiliary $|0\rangle$-qubit to

$$\sqrt{1 - \frac{1}{\kappa^2 \lambda_j^2}}|0\rangle + \frac{1}{\kappa\lambda_j}|1\rangle$$

(this is a valid state because $|\kappa\lambda_j| \geqslant 1$).

§8.2 The Basic HHL Algorithm for Linear Systems

Unfortunately the maps A and A^{-1} are not unitary (unless $|\lambda_j| = 1$ for all j), so we cannot just apply A^{-1} as a quantum operation to state $|b\rangle$ to get state $|x\rangle$. Fortunately $U = e^{iA} = \sum_{j=0}^{N-1} e^{i\lambda_j} a_j a_j^\dagger$ is unitary, and has the same eigenvectors as A and A^{-1}. We can implement U and powers of U by Hamiltonian simulation, and then use phase estimation to estimate the λ_j associated with eigenvector $|a_j\rangle$ with some small approximation error.

Conditioned on our estimate of λ_j we can then rotate an auxiliary $|0\rangle$-qubit to

$$\sqrt{1 - \frac{1}{\kappa^2 \lambda_j^2}}|0\rangle + \frac{1}{\kappa\lambda_j}|1\rangle$$

(this is a valid state because $|\kappa\lambda_j| \geqslant 1$).

§8.2 The Basic HHL Algorithm for Linear Systems

Next we undo the phase estimation to set the register that contained the estimate back to $|0\rangle$. Suppressing the auxiliary qubits containing the temporary results of the phase estimation, we have now unitarily mapped

$$|a_j\rangle|0\rangle \mapsto |a_j\rangle \otimes \left(\sqrt{1-\frac{1}{\kappa^2\lambda_j^2}}|0\rangle + \frac{1}{\kappa\lambda_j}|1\rangle\right).$$

If we prepare a copy of $|b\rangle|0\rangle = \sum_{j=0}^{N-1} \beta_j|a_j\rangle|0\rangle$ and apply the above unitary map to it, then we obtain

$$\sum_{j=0}^{N-1}\beta_j|a_j\rangle\left(\sqrt{1-\frac{1}{\kappa^2\lambda_j^2}}|0\rangle + \frac{1}{\kappa\lambda_j}|1\rangle\right) = |\phi\rangle|0\rangle + \frac{1}{\kappa}\sum_{j=0}^{N-1}\beta_j\frac{1}{\lambda_j}|a_j\rangle|1\rangle,$$

where we do not care about the (sub-normalized) state $|\phi\rangle$.

§8.2 The Basic HHL Algorithm for Linear Systems

Next we undo the phase estimation to set the register that contained the estimate back to $|0\rangle$. Suppressing the auxiliary qubits containing the temporary results of the phase estimation, we have now unitarily mapped

$$|a_j\rangle|0\rangle \mapsto |a_j\rangle \otimes \left(\sqrt{1-\frac{1}{\kappa^2\lambda_j^2}}|0\rangle + \frac{1}{\kappa\lambda_j}|1\rangle\right).$$

If we prepare a copy of $|b\rangle|0\rangle = \sum_{j=0}^{N-1} \beta_j|a_j\rangle|0\rangle$ and apply the above unitary map to it, then we obtain

$$\sum_{j=0}^{N-1}\beta_j|a_j\rangle\left(\sqrt{1-\frac{1}{\kappa^2\lambda_j^2}}|0\rangle + \frac{1}{\kappa\lambda_j}|1\rangle\right) = |\phi\rangle|0\rangle + \frac{1}{\kappa}\sum_{j=0}^{N-1}\beta_j\frac{1}{\lambda_j}|a_j\rangle|1\rangle,$$

where we do not care about the (sub-normalized) state $|\phi\rangle$.

§8.2 The Basic HHL Algorithm for Linear Systems

Note that because $\sum_{j=0}^{N-1} |\beta_j/\lambda_j|^2 \geqslant \sum_{j=0}^{N-1} |\beta_j|^2 = 1$, the norm of the part of the state ending in qubit $|0\rangle$ is at least $1/\kappa^2$. Accordingly, we can now apply $\mathcal{O}(\kappa)$ rounds of amplitude amplification to amplify this part of the state to have amplitude essentially 1. This prepares state $|x\rangle$, as intended. This rough sketch is the basic idea of HHL. It leads to an algorithm that produces a state $|\widetilde{x}\rangle$ that is ε-close to $|x\rangle$, using roughly $\kappa^2 s/\varepsilon$ queries to H and roughly $\kappa s(\kappa n/\varepsilon + B)$ other 2-qubit gates.

§8.2 The Basic HHL Algorithm for Linear Systems

Note that because $\sum_{j=0}^{N-1} |\beta_j/\lambda_j|^2 \geqslant \sum_{j=0}^{N-1} |\beta_j|^2 = 1$, the norm of the part of the state ending in qubit $|0\rangle$ is at least $1/\kappa^2$. Accordingly, we can now apply $\mathcal{O}(\kappa)$ rounds of amplitude amplification to amplify this part of the state to have amplitude essentially 1. This prepares state $|x\rangle$, as intended. This rough sketch is the basic idea of HHL. It leads to an algorithm that produces a state $|\widetilde{x}\rangle$ that is ε-close to $|x\rangle$, using roughly $\kappa^2 s/\varepsilon$ queries to H and roughly $\kappa s(\kappa n/\varepsilon + B)$ other 2-qubit gates.

§8.2 The Basic HHL Algorithm for Linear Systems

Note that because $\sum_{j=0}^{N-1} |\beta_j/\lambda_j|^2 \geqslant \sum_{j=0}^{N-1} |\beta_j|^2 = 1$, the norm of the part of the state ending in qubit $|0\rangle$ is at least $1/\kappa^2$. Accordingly, we can now apply $\mathcal{O}(\kappa)$ rounds of amplitude amplification to amplify this part of the state to have amplitude essentially 1. This prepares state $|x\rangle$, as intended. This rough sketch is the basic idea of HHL. It leads to an algorithm that produces a state $|\widetilde{x}\rangle$ that is ε-close to $|x\rangle$, using roughly $\kappa^2 s/\varepsilon$ queries to H and roughly $\kappa s(\kappa n/\varepsilon + B)$ other 2-qubit gates.

§8.2 The Basic HHL Algorithm for Linear Systems

§8.2.1 Illustration of the quantum circuits for HHL

The algorithm uses three quantum registers, all of them set to $|0\rangle$ at the beginning of the algorithm. One register, which we will denote with the sub-index n_ℓ, is used to store a binary representation of the eigenvalues of A. A second register, denoted by n_b, contains the vector solution, and from now on $N = 2^{n_b}$. There is an extra register, for the auxiliary qubits. These are qubits used as intermediate steps in the individual computations but will be ignored in the following description since they are set to $|0\rangle$ at the beginning of each computation and restored back to the $|0\rangle$ state at the end of the individual operation.

§8.2 The Basic HHL Algorithm for Linear Systems

§8.2.1 Illustration of the quantum circuits for HHL

The algorithm uses three quantum registers, all of them set to $|0\rangle$ at the beginning of the algorithm. One register, which we will denote with the sub-index n_ℓ, is used to store a binary representation of the eigenvalues of A. A second register, denoted by n_b, contains the vector solution, and from now on $N = 2^{n_b}$. There is an extra register, for the auxiliary qubits. These are qubits used as intermediate steps in the individual computations but will be ignored in the following description since they are set to $|0\rangle$ at the beginning of each computation and restored back to the $|0\rangle$ state at the end of the individual operation.

§8.2 The Basic HHL Algorithm for Linear Systems

§8.2.1 Illustration of the quantum circuits for HHL

The algorithm uses three quantum registers, all of them set to $|0\rangle$ at the beginning of the algorithm. One register, which we will denote with the sub-index n_ℓ, is used to store a binary representation of the eigenvalues of A. A second register, denoted by n_b, contains the vector solution, and from now on $N = 2^{n_b}$. There is an extra register, for the auxiliary qubits. These are qubits used as intermediate steps in the individual computations but will be ignored in the following description since they are set to $|0\rangle$ at the beginning of each computation and restored back to the $|0\rangle$ state at the end of the individual operation.

§8.2 The Basic HHL Algorithm for Linear Systems

§8.2.1 Illustration of the quantum circuits for HHL

The algorithm uses three quantum registers, all of them set to $|0\rangle$ at the beginning of the algorithm. One register, which we will denote with the sub-index n_ℓ, is used to store a binary representation of the eigenvalues of A. A second register, denoted by n_b, contains the vector solution, and from now on $N = 2^{n_b}$. There is an extra register, for the auxiliary qubits. These are qubits used as intermediate steps in the individual computations but will be ignored in the following description since they are set to $|0\rangle$ at the beginning of each computation and restored back to the $|0\rangle$ state at the end of the individual operation.

§8.2 The Basic HHL Algorithm for Linear Systems

The following is an outline of the HHL algorithm with a **high-level drawing** of the corresponding circuit. For simplicity all computations are assumed to be exact in the ensuing description.

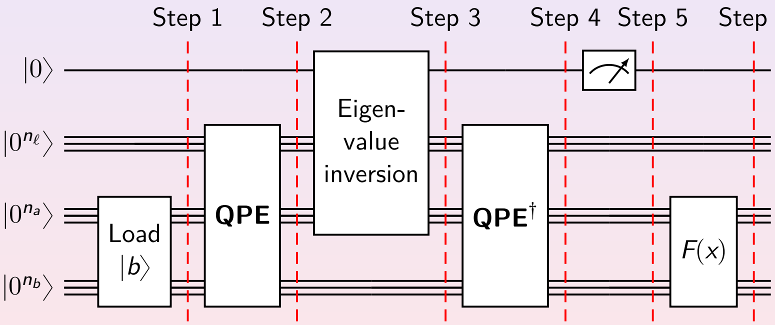

Figure 1: The quantum circuit of the HHL algorithm

§8.2 The Basic HHL Algorithm for Linear Systems

Step 1: Load the data $|b\rangle \in \mathbb{C}^N$; that is, perform the transformation $|0^{n_b}\rangle \mapsto |b\rangle$.

Step 2: Apply Quantum Phase Estimation (QPE) with

$$U = e^{iAt} = \sum_{j=0}^{N-1} e^{i\lambda_j t}|a_j\rangle\langle a_j|$$

for a certain t (here we take $t = 1$). The quantum state of the register expressed in the eigenbasis of A is now $\sum_{j=0}^{N-1} \beta_j|\lambda_j\rangle_{n_\ell}|a_j\rangle$; that is,

$$\mathbf{QPE}(U, |0^{n_\ell}\rangle|b\rangle) = \sum_{j=0}^{N-1} \beta_j|\lambda_j\rangle_{n_\ell}|a_j\rangle .$$

Here we recall that $|\lambda_j\rangle_{n_\ell}$ is the n_ℓ-bit binary approximation of λ_j satisfying $|\lambda_j\rangle_{n_\ell} = |[2^{n_\ell}\lambda_j]\rangle$.

§8.2 The Basic HHL Algorithm for Linear Systems

Step 1: Load the data $|b\rangle \in \mathbb{C}^N$; that is, perform the transformation $|0^{n_b}\rangle \mapsto |b\rangle$.

Step 2: Apply Quantum Phase Estimation (QPE) with

$$U = e^{iAt} = \sum_{j=0}^{N-1} e^{i\lambda_j t} |a_j\rangle\langle a_j|$$

for a certain t (here we take $t = 1$). The quantum state of the register expressed in the eigenbasis of A is now $\sum_{j=0}^{N-1} \beta_j |\lambda_j\rangle_{n_\ell} |a_j\rangle$; that is,

$$\mathbf{QPE}(U, |0^{n_\ell}\rangle|b\rangle) = \sum_{j=0}^{N-1} \beta_j |\lambda_j\rangle_{n_\ell} |a_j\rangle\,.$$

Here we recall that $|\lambda_j\rangle_{n_\ell}$ is the n_ℓ-bit binary approximation of λ_j satisfying $|\lambda_j\rangle_{n_\ell} = |[2^{n_\ell}\lambda_j]\rangle$.

§8.2 The Basic HHL Algorithm for Linear Systems

Step 3: Add an auxiliary qubit and apply a rotation conditioned on $|\lambda_j\rangle_{n_\ell}$ (multi-controlled rotation gates),

$$\sum_{j=0}^{N-1} \beta_j |\lambda_j\rangle_{n_\ell} |a_j\rangle \left(\sqrt{1 - \frac{1}{\kappa^2 \lambda_j^2}} |0\rangle + \frac{1}{\kappa \lambda_j} |1\rangle \right),$$

where κ is (an upper bound of) the condition number of A.

Step 4: Apply $\mathrm{QPE}^\dagger$ (that is, undo **QPE**). Ignoring possible errors from **QPE**, this results in

$$\sum_{j=0}^{N-1} \beta_j |0^{n_\ell}\rangle |a_j\rangle \left(\sqrt{1 - \frac{1}{\kappa^2 \lambda_j^2}} |0\rangle + \frac{1}{\kappa \lambda_j} |1\rangle \right).$$

§8.2 The Basic HHL Algorithm for Linear Systems

Step 3: Add an auxiliary qubit and apply a rotation conditioned on $|\lambda_j\rangle_{n_\ell}$ (multi-controlled rotation gates),

$$\sum_{j=0}^{N-1} \beta_j |\lambda_j\rangle_{n_\ell} |a_j\rangle \left(\sqrt{1 - \frac{1}{\kappa^2 \lambda_j^2}} |0\rangle + \frac{1}{\kappa \lambda_j} |1\rangle \right),$$

where κ is (an upper bound of) the condition number of A.

Step 4: Apply $\mathbf{QPE}^\dagger$ (that is, undo **QPE**). Ignoring possible errors from **QPE**, this results in

$$\sum_{j=0}^{N-1} \beta_j |0^{n_\ell}\rangle |a_j\rangle \left(\sqrt{1 - \frac{1}{\kappa^2 \lambda_j^2}} |0\rangle + \frac{1}{\kappa \lambda_j} |1\rangle \right).$$

§8.2 The Basic HHL Algorithm for Linear Systems

Step 5: Measure the auxiliary qubit in the computational basis. If the outcome is 1, the register is in the post-measurement state

$$\left(\frac{1}{\sum_{j=0}^{N-1}|\beta_j|^2|\lambda_j|^{-2}}\right)^{\frac{1}{2}}\sum_{j=0}^{N-1}\frac{\beta_j}{\lambda_j}|0^{n_\ell}\rangle|a_j\rangle$$

which up to a normalisation factor corresponds to the solution.

Step 6: Apply an observable M to calculate $F(x) \equiv \langle x|M|x\rangle$.

§8.2 The Basic HHL Algorithm for Linear Systems

Step 5: Measure the auxiliary qubit in the computational basis. If the outcome is 1, the register is in the post-measurement state

$$\left(\frac{1}{\sum_{j=0}^{N-1}|\beta_j|^2|\lambda_j|^{-2}}\right)^{\frac{1}{2}}\sum_{j=0}^{N-1}\frac{\beta_j}{\lambda_j}|0^{n_\ell}\rangle|a_j\rangle$$

which up to a normalisation factor corresponds to the solution.

Step 6: Apply an observable M to calculate $F(x) \equiv \langle x|M|x\rangle$.

§8.2 The Basic HHL Algorithm for Linear Systems

Example

Consider solving the linear system $Ax = b$, where

$$A = \begin{bmatrix} 1 & -1/3 \\ -1/3 & 1 \end{bmatrix} \quad \text{and} \quad |b\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

We will use $n_b = 1$ qubit to represent $|b\rangle$ and later the solution $|x\rangle$, $n_\ell = 2$ qubits to store the binary representation of the eigenvalues, and 1 auxiliary qubit to store whether the conditioned rotation, hence the algorithm, was successful.

For the purpose of illustrating the algorithm, we will cheat a bit and calculate the eigenvalues of A to be able to choose t to obtain an exact binary representation of the rescaled eigenvalues in the n_ℓ-register. However, keep in mind that for the HHL algorithm implementation one does not need knowledge of the eigenvalues.

§8.2 The Basic HHL Algorithm for Linear Systems

Example

Consider solving the linear system $Ax = b$, where

$$A = \begin{bmatrix} 1 & -1/3 \\ -1/3 & 1 \end{bmatrix} \quad \text{and} \quad |b\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

We will use $n_b = 1$ qubit to represent $|b\rangle$ and later the solution $|x\rangle$, $n_\ell = 2$ qubits to store the binary representation of the eigenvalues, and 1 auxiliary qubit to store whether the conditioned rotation, hence the algorithm, was successful.

For the purpose of illustrating the algorithm, we will cheat a bit and calculate the eigenvalues of A to be able to choose t to obtain an exact binary representation of the rescaled eigenvalues in the n_ℓ-register. However, keep in mind that for the HHL algorithm implementation one does not need knowledge of the eigenvalues.

§8.2 The Basic HHL Algorithm for Linear Systems

Example (cont.)

Recall that the **QPE** will output an n_ℓ-bit (2-bit in this case) binary approximation to $2^n\lambda_j t$. Since the eigenvalues of A are $\lambda_1 = 2/3$ and $\lambda_2 = 4/3$, if we set $t = \dfrac{3\pi}{4}$ the **QPE** will give a 2-bit binary approximation to $\dfrac{\lambda_1 t}{2\pi} = \dfrac{1}{4}$ and $\dfrac{\lambda_2 t}{2\pi} = \dfrac{1}{2}$, which is, respectively,

$$|01\rangle_{n_\ell} \quad \text{and} \quad |10\rangle_{n_\ell}.$$

The eigenvectors are, respectively,

$$|a_1\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \text{and} \quad |a_2\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Again, keep in mind that one does not need to compute the eigenvectors for the HHL implementation.

§8.2 The Basic HHL Algorithm for Linear Systems

Example (cont.)

Recall that the **QPE** will output an n_ℓ-bit (2-bit in this case) binary approximation to $2^n\lambda_j t$. Since the eigenvalues of A are $\lambda_1 = 2/3$ and $\lambda_2 = 4/3$, if we set $t = \dfrac{3\pi}{4}$ the **QPE** will give a 2-bit binary approximation to $\dfrac{\lambda_1 t}{2\pi} = \dfrac{1}{4}$ and $\dfrac{\lambda_2 t}{2\pi} = \dfrac{1}{2}$, which is, respectively,

$$|01\rangle_{n_\ell} \qquad \text{and} \qquad |10\rangle_{n_\ell}.$$

The eigenvectors are, respectively,

$$|a_1\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad \text{and} \qquad |a_2\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Again, keep in mind that one does not need to compute the eigenvectors for the HHL implementation.

§8.2 The Basic HHL Algorithm for Linear Systems

Example (cont.)

We can then write $|b\rangle$ in the eigenbasis of A as

$$|b\rangle = \sum_{j=1}^{2} \frac{1}{\sqrt{2}} |a_j\rangle .$$

Now we are ready to go through the different steps of the HHL algorithm.

Step 1: State preparation in this example is trivial since $|b\rangle = |0\rangle$.

Step 2: Applying QPE will yield

$$\frac{1}{\sqrt{2}}|01\rangle|a_1\rangle + \frac{1}{\sqrt{2}}|10\rangle|a_2\rangle .$$

§8.2 The Basic HHL Algorithm for Linear Systems

Example (cont.)

Step 3: Conditioned rotation with $\kappa = 8$. Note, the constant κ here needs to be chosen such that the product of κ and the smallest eigenvalue $\frac{1}{4}$ is bigger than 1 but as small as possible so that when the auxiliary qubit is measured, the probability of it being in the state $|1\rangle$ is large:

$$\begin{aligned}
&\frac{1}{\sqrt{2}}|01\rangle|a_1\rangle\left(\sqrt{1-\frac{1}{8^2\cdot 1/4^2}}|0\rangle+\frac{1}{8\cdot 1/4}|1\rangle\right)\\
&\quad+\frac{1}{\sqrt{2}}|10\rangle|a_2\rangle\left(\sqrt{1-\frac{1}{8^2\cdot 1/2^2}}|0\rangle+\frac{1}{8\cdot 1/2}|1\rangle\right)\\
&\qquad=\frac{1}{\sqrt{2}}|01\rangle|a_1\rangle\left(\frac{\sqrt{3}}{2}|0\rangle+\frac{1}{2}|1\rangle\right)+\frac{1}{\sqrt{2}}|10\rangle|a_2\rangle\left(\frac{\sqrt{15}}{4}|0\rangle+\frac{1}{4}|1\rangle\right).
\end{aligned}$$

§8.2 The Basic HHL Algorithm for Linear Systems

Example (cont.)

Step 4: After applying $\mathbf{QPE}^\dagger$ the quantum computer is in the state

$$\frac{1}{\sqrt{2}}|00\rangle|a_1\rangle\left(\frac{\sqrt{3}}{2}|0\rangle+\frac{1}{2}|1\rangle\right)+\frac{1}{\sqrt{2}}|00\rangle|a_2\rangle\left(\frac{\sqrt{15}}{4}|0\rangle+\frac{1}{4}|1\rangle\right).$$

Step 5: On outcome 1 when measuring the auxiliary qubit, the state is

$$\frac{1}{\sqrt{5/32}}\left(\frac{1}{\sqrt{2}}|00\rangle|a_1\rangle\frac{1}{2}|1\rangle+\frac{1}{\sqrt{2}}|00\rangle|a_2\rangle\frac{1}{4}|1\rangle\right).$$

A quick calculation shows that

$$\frac{1}{\sqrt{5/32}}\left(\frac{1}{2\sqrt{2}}|a_1\rangle+\frac{1}{4\sqrt{2}}|a_2\rangle\right)=\frac{x}{\|x\|},$$

where $x=\begin{bmatrix}9/8 & 3/8\end{bmatrix}^{\mathrm{T}}$ is the solution.

§8.2 The Basic HHL Algorithm for Linear Systems

Example (cont.)

Step 4: After applying $\mathbf{QPE}^\dagger$ the quantum computer is in the state

$$\frac{1}{\sqrt{2}}|00\rangle|a_1\rangle\left(\frac{\sqrt{3}}{2}|0\rangle+\frac{1}{2}|1\rangle\right)+\frac{1}{\sqrt{2}}|00\rangle|a_2\rangle\left(\frac{\sqrt{15}}{4}|0\rangle+\frac{1}{4}|1\rangle\right).$$

Step 5: On outcome 1 when measuring the auxiliary qubit, the state is

$$\frac{1}{\sqrt{5/32}}\left(\frac{1}{\sqrt{2}}|00\rangle|a_1\rangle\frac{1}{2}|1\rangle+\frac{1}{\sqrt{2}}|00\rangle|a_2\rangle\frac{1}{4}|1\rangle\right).$$

A quick calculation shows that

$$\frac{1}{\sqrt{5/32}}\left(\frac{1}{2\sqrt{2}}|a_1\rangle+\frac{1}{4\sqrt{2}}|a_2\rangle\right)=\frac{x}{\|x\|},$$

where $x=\begin{bmatrix}9/8 & 3/8\end{bmatrix}^{\mathrm{T}}$ is the solution.

§8.2 The Basic HHL Algorithm for Linear Systems

Example (cont.)

Step 6: Without using extra gates, we can compute the norm of $|x\rangle$: it is the probability of measuring 1 in the auxiliary qubit from the previous step

$$P(|1\rangle) = \left(\frac{1}{2\sqrt{2}}\right)^2 + \left(\frac{1}{4\sqrt{2}}\right)^2 = \frac{5}{32}\,.$$