

NUMERICAL ANALYSIS ON QUANTUM COMPUTERS

MARKUS FAUSTMANN, MICHAEL FEISCHL

CONTENTS

1. Introduction	3
1.1. Short history of quantum computing	3
1.2. Aim of this lecture	3
1.3. Quantum Mechanics	4
2. Mathematical basics of quantum computing	5
2.1. Dirac Notation (Bra-Ket-Notation)	5
2.2. Digression: Tensor product of Hilbert spaces	6
2.3. Measurement	9
3. Fundamental operations on quantum computers	11
3.1. Quantum circuits	11
3.2. Single qubit gates	12
3.3. Two qubit gates	12
3.4. Three qubit gates	13
3.5. Quantum information processing	15
3.6. Encoding	17
3.7. Complexity	19
4. Classical Quantum Algorithms	20
4.1. Deutsch-Josza-Algorithm	20
4.2. Simon's algorithm	25
4.3. The quantum Fourier Transform	27
4.4. Application of QFT: Phase Estimation	29
4.5. Shor's integer factorization	31
4.6. Grover's search algorithm	35
5. Numerical algorithms on quantum computers	42
5.1. Numerical Integration	42
5.2. Solving linear systems of equations	44
5.3. Hamiltonian simulation	47
5.4. Graph coloring method	49
5.5. Modification for 3-diagonal matrices	53
5.6. Loading the right hand side	53
5.7. HHL revisited	55
5.8. Error and complexity analysis	58
5.9. Improvements on the HHL-algorithm	60
5.10. Newton's method on quantum computers	66
5.11. Solving Linear Differential Equations	66
5.12. Multi-step methods	68

1. INTRODUCTION

1.1. Short history of quantum computing. Quantum Computing is a fundamentally new paradigm to information processing on a computing device. While classical computers use bits with logical values 0 and 1 as smallest units of information, quantum computers use so called qubits (a notion mathematically introduced in the next chapter), which can not only take states 0 and 1 but also linear combinations of these two groundstates.

Quantum computers are devices that manipulate qubits based on principles of quantum mechanics to make calculations. Quantum mechanics is a field in physics that aims to describe behaviors in sub-atomic scales and historically dates back to the 1920-1930s and is a highly unintuitive field, as demonstrated by Erwin Schrödingers thought experiment of "Schrödingers cat", which demonstrates that one can not decide whether the cat in the experiment is dead or alive and thus it is in a superposition of both.

The idea of using devices based on principles of quantum mechanics for computing purposes dates back to the 1980s, with e.g. early works by Nobel prize winner Richard Feynman and a formal description of a quantum Turing machine by David Deutsch in 1985. There, the main goal was to actually reduce the physics in quantum computing to a bare minimum of rules (the axioms of quantum computing, introduced in the following section).

Using the power of qubits and superpositions of states, in the 1990s the first algorithms (also presented in Chapter 4) were developed that allow - in theory - for exponential speed up computations in comparison with classical computers. The key breakthrough hereby came in 1994 when mathematician Peter Shor proved that the integer factorization problem, a key to our current cryptography algorithms, can be efficiently solved on quantum computers.

While the theoretical prowess of quantum computers is indisputable, concrete and useful realizations of quantum computers are a topic of ongoing research. The currently dominant architecture is superconductor based qubits, where ground states are low and high energy levels and manipulations are done via microwaves. The biggest issue right now is that these qubits are heavily affected by noise and can only hold there states for milliseconds. However, in 2025 the first quantum computer based on this architecture with over 1000 qubits was built and real calculations on quantum computers can be made nowadays. Even though many claims have been made that quantum supremacy, i.e., calculations on real quantum computers for certain tailored problems with exponential speed up compared to classical computers, has been reached, advantages in real, useful calculations have not been done, undisputably, yet.

1.2. Aim of this lecture. The aim of this lecture is to give a short introduction into the mathematic description of quantum computing, most notably by introducing the axioms of quantum computing and so called gates for manipulation of quantum states. Then, the main focus of the lecture will be laid onto quantum algorithms, i.e., algorithms that allow for exponential speed ups, which will be rigorously proven for important problems such as integer factorization and unstructured search.

Then, we visit classical problems in numerical mathematics such as numerical integration, solution of linear systems of equations and and numerical solution of differential equations. Hereby, we discuss how they can be formulated on quantum computers and whether they can be solved - in theory - with exponential speed up.

In general, this lecture notes looks at quantum computing from a numerical analysts view and tries to give an idea how quantum algorithms can be analyzed in terms of theoretical quantum advantages. For some of the algorithms, we also provide, as supplemental materials, implementations in the IBM quantum composer or the programming language qiskit.

This is version 1 of the lecture notes written in ST2026 at TU Wien and we are very grateful by the help of Sebastian Hirnschall in writing the initial latex version. The material in the lecture notes is based on various different scientific articles and lecture notes in the field of quantum computing. As overview articles, we can recommend [3, 5].

1.3. Quantum Mechanics. In the same way as the pioneers of quantum computing, we do not aim to provide any explanation of quantum mechanics, but rather only use rules imposed by quantum mechanics.

In a nutshell, the evolution of every quantum system ψ , and thus also every quantum computer can be described by a Schrödinger equation

$$\psi'(t) = -iH\psi(t),$$

governed by its Hamiltonian H . Formally, a solution reads

$$\psi(t) = \underbrace{e^{-iHt}}_{U(t)\dots\text{unitary operator}} \psi(0)$$

and thus is nothing else but an evolution of the initial state $\psi(0)$ under an unitary operator. The description by means of Schrödinger equations acts as a direct motivation for the axioms of quantum computing in the following section.

2. MATHEMATICAL BASICS OF QUANTUM COMPUTING

In this section, we give a short introduction into the basics of quantum computing, where we most notably introduce qubits as a basic unit of information.

We start with the previously mentioned rules for computation imposed by quantum mechanics.

(QM1): A state of an (isolated) quantum system is described by a unit vector ($\|\cdot\|_H = 1$) in a complex valued Hilbert space H . The space H is called state space.

2.1. Dirac Notation (Bra-Ket-Notation). Let H be a complex Hilbert space with $v \in H$. In Dirac's notation, one writes $|v\rangle := v$ called *ket-vector*.

If the Hilbert space H is finite dimensional with $N := \dim H < \infty$, we may choose an orthonormal basis of H denoted by $\{|0\rangle, |1\rangle, \dots, |N-1\rangle\}$. A quantum state $|\Phi\rangle \in H$ can then be written a superposition of basis states

$$|\Phi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle + \dots + \alpha_{N-1}|N-1\rangle$$

where $\alpha_i \in \mathbb{C}$ is called the amplitude of the basis state $|i\rangle$ in $|\Phi\rangle$. We can therefore interpret

$$|\Phi\rangle = \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{N-1} \end{pmatrix}$$

as vector in \mathbb{C}^N . A *bra* vector $\langle\Phi|$ is the conjugate transpose of $|\Phi\rangle$, i.e.,

$$\langle\Phi| := |\Phi\rangle^H = (\bar{\alpha}_0, \bar{\alpha}_1, \dots, \bar{\alpha}_{N-1}),$$

and the inner product of two states $|\Phi\rangle$ and $|\Psi\rangle$ can conveniently be written as

$$\langle\Phi|\Psi\rangle := \langle\Phi, \Psi\rangle = \langle\Phi| \cdot |\Psi\rangle.$$

Example 1. For $N = 2$ we have $|\Phi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ with $|\alpha_0|^2 + |\alpha_1|^2 = 1$.

Definition 2. A quantum system with 2-dimensional state space (ONB $|0\rangle, |1\rangle$) is called *qubit*.

As motivated by the Schrödinger equation above, changes in a quantum system are done via unitary operators, which we formulate as the second axiom of quantum computing.

(QM2): The evolution of a quantum system is only described by unitary operations acting on the state space.

Note that the unitary evolution of a qubit $|\Phi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$, $|\alpha_0|^2 + |\alpha_1|^2 = 1$ by

$$U(|\Phi\rangle) = |\psi\rangle = \beta_0|0\rangle + \beta_1|1\rangle$$

implies that $|\beta_0|^2 + |\beta_1|^2 = 1$ as unitary operators conserve lengths. Thus, we obtain that $U(|\Phi\rangle)$ is a qubit as well.

In order to be able to make computations, it is important to combine quantum systems, which leads to the third axiom of quantum computing.

(QM3): Let S_1, S_2 be quantum systems with state spaces V, W . The composition of S_1 and S_2 is described by the tensor product $V \otimes W$.

2.2. Digression: Tensor product of Hilbert spaces. In the following, for a pair of Hilbert spaces V, W , we formally define the tensor product $V \otimes W$.

Definition 3. Let $(V, (\cdot, \cdot)_V), (W, (\cdot, \cdot)_W)$ be complex Hilbert spaces. A complex Hilbert space $(H, (\cdot, \cdot)_H)$ together with a bilinear mapping $\omega_H : V \times W \rightarrow H$ is called tensor product of V, W if

- $\text{span}\{\omega_H(v, w) : v \in V, w \in W\}$ is dense in H and
- there holds

$$(\omega_H(v_1, w_1), \omega_H(v_2, w_2))_H = (v_1, v_2)_V \cdot (w_1, w_2)_W \quad (1)$$

for all $v_1, v_2 \in V, w_1, w_2 \in W$.

There are many ways to show the existence of the tensor product. Here, we sketch two possibilities:

- A simple way is to define a tensor product of complex valued square summable series $\ell^2(M_i)$ on (index) sets $M_i, i = 1, 2$ (understood as functions $M_i \rightarrow \mathbb{C}$) as $\ell^2(M_1 \times M_2)$ with $\omega_{\ell^2}(g_1, g_2)(m_1, m_2) := g_1(m_1)g_2(m_2)$.

Now, by ONB basis expansion, one obtains a mapping $\mathcal{F}_j : H_j \rightarrow \ell^2(M_j)$ mapping an element in the Hilbert space to the sequence of Fourier coefficients. Consequently, the mapping $\omega_{\ell^2} \circ (\mathcal{F}_1 \times \mathcal{F}_2) : V \times W \rightarrow \ell^2(M_1 \times M_2)$ satisfies all conditions for a tensor product. For more details, we refer to [4].

- A more direct and constructive way is to start with the product space $V \times W$ and construct a vector space that inherits the vector space structure of both V, W . More precisely, define the free vector space $\mathcal{F}(V, W)$ as the set of all finite linear combinations of elements of $V \times W$, i.e.,

$$\mathcal{F}(V, W) = \left\{ \sum_{j=1}^n \alpha_j (v_j, w_j) : v_j \in V, w_j \in W, \alpha_j \in \mathbb{C} \right\}.$$

Defining the non-empty subspace

$$U(V, W) = \text{span} \left\{ \sum_{j,k=1}^n \alpha_j \beta_k (v_j, w_j) - \left(\sum_{j=1}^n \alpha_j v_j, \sum_{k=1}^n \beta_k w_k \right) \right\} \subseteq \mathcal{F}(V, W)$$

induces an equivalence relation \sim given by

$$(v_1, w_1) \sim (v_2, w_2) \iff (v_1, w_1) - (v_2, w_2) \in U(V, W).$$

Now, on the space $\mathcal{F}(V, W)/\sim$, i.e., equivalence classes of $\mathcal{F}(V, W)$ under \sim , one can define an inner product (to show definiteness it is important to work with equivalence classes)

$$\langle v_1 \otimes w_1, v_2 \otimes w_2 \rangle_H := \langle v_1, v_2 \rangle_V \langle w_1, w_2 \rangle_W.$$

Finally, the tensor product space H is the completion of $\mathcal{F}(V, W)/\sim$ under the norm $\|\cdot\|_H := \langle \cdot, \cdot \rangle_H$, which actually satisfies all conditions of a tensor product. For more details, we refer to [6].

An immediate and very useful consequence of (1) is that ONBs for the spaces V, W induce an ONB for a tensor product.

Lemma 4. Let $E_V \subset V$ be an ONB of V and $E_W \subset W$ be an ONB of W . Then, $\omega_H(E_V \times E_W)$ is an ONB of the tensor product H, ω_H of V, W .

Proof. The orthonormality of $\omega_H(E_V \times E_W)$ follows from (1) and the fact that E_V, E_W are orthonormal sets.

The mapping $\omega_H : V \times W \rightarrow H$ is continuous, since, for $(v_1, w_1), (v_2, w_2) \in V \times W$ with $\|(v_1, w_1)\|_{V \times W}, \|(v_2, w_2)\|_{V \times W} \leq C$, we have

$$\begin{aligned} \|\omega_H(v_1, w_1) - \omega_H(v_2, w_2)\|_H &= \|\omega_H(v_1 - v_2, w_1) + \omega_H(v_2, w_1 - w_2)\|_H \\ &\leq \|\omega_H(v_1 - v_2, w_1)\|_H + \|\omega_H(v_2, w_1 - w_2)\|_H \\ &\leq \|v_1 - v_2\|_V \|w_1\|_W + \|v_2\|_V \|w_1 - w_2\|_W \\ &\leq C \|(v_1, w_1) - (v_2, w_2)\|_{V \times W}. \end{aligned}$$

Together with bilinearity of ω_H this implies

$$\begin{aligned} \text{span } \omega_H(V \times W) &= \text{span } \omega_H(\overline{\text{span } E_V} \times \overline{\text{span } E_W}) = \text{span } \omega_H(\overline{\text{span } E_V \times \text{span } E_W}) \\ &\subseteq \overline{\text{span } \omega_H(\text{span } E_V \times \text{span } E_W)} = \overline{\text{span } \omega_H(E_V \times E_W)}. \end{aligned}$$

As the left-hand side is dense in H , so is the right-hand side and thus $\omega_H(E_V \times E_W)$ is an ONB of H . \square

The tensor product of two Hilbert spaces is unique up to a unitary transformation.

Lemma 5. *Let H, ω_H and G, ω_G be two tensor products of the Hilbert spaces V, W . Then, there exists a unitary operator $U : H \rightarrow G$ with $U \circ \omega_H = \omega_G$.*

Proof. By density, it is enough to show the result for an orthonormal basis. Let $E_V \subset V$ be an ONB and $E_W \subset W$ be an ONB. By the previous lemma, we have that $\omega_H(E_V \times E_W)$ is an ONB of H . The same thing holds for $\omega_G(E_V \times E_W)$ and G . Thus, there exists a unitary operator $U : G \rightarrow H$ such that

$$U(\omega_H(e_V, e_W)) = \omega_G(e_V, e_W) \quad \forall e_V \in E_V, e_W \in E_W.$$

Now, continuity and multilinearity implies $U \circ \omega_H = \omega_G$. \square

Notation: By existence and uniqueness of the tensor product, we may write $V \otimes W$ instead of H and $v \otimes w$ instead of $\omega_H(v, w)$. By bilinearity of the tensor product, we have

$$\begin{aligned} (v_1 + v_2) \otimes w &= v_1 \otimes w + v_2 \otimes w \\ v \otimes (w_1 + w_2) &= v \otimes w_1 + v \otimes w_2 \\ \alpha(v \otimes w) &= (\alpha v) \otimes w + v \otimes (\alpha w). \end{aligned}$$

Remark 6. *Note that:*

- Lemma 4 implies that $\dim(V \otimes W) = \dim V \cdot \dim W$.
- Up to unitary transformation, the tensor product of two spaces is commutative, i.e., $V \otimes W = W \otimes V$.
- The construction/definition of the tensor product can easily be generalized for a product of finitely many Hilbert spaces, i.e., $V_1 \otimes V_2 \otimes \cdots \otimes V_n$ by requiring that the function $\omega_H : V_1 \times V_2 \times \cdots \times V_n \rightarrow H$ in the definition is multilinear instead of bilinear and modifying (1) accordingly.

Example 7. (1) *For real/complex matrices the tensor product is given by the Kronecker product: Let $A \in \mathbb{K}^{n \times m}$, $B \in \mathbb{K}^{k \times \ell}$, then the tensor product is given by*

$$A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1m}B \\ \vdots & & \vdots \\ a_{n1}B & \cdots & a_{nm}B \end{pmatrix} \in \mathbb{K}^{nk \times m\ell}.$$

As a special case, we obtain that $\mathbb{R}^2 \otimes \mathbb{R}^2$ is 4-dimensional with

$$\begin{aligned} \mathbb{R}^2 \otimes \mathbb{R}^2 &= \text{span} \left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right\} \\ &= \text{span} \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \right\} = \mathbb{R}^4. \end{aligned}$$

(2) Let H be a Hilbert space then

$$H \otimes \mathbb{C}^n = H^n = H \times \dots \times H.$$

(3) Let (M_1, μ_1) and (M_2, μ_2) be measure spaces. Then, by Fubini's theorem, one can show that the tensor product is

$$L^2(M_1, \mu_1) \otimes L^2(M_2, \mu_2) = L^2(M_1 \times M_2, \mu_1 \otimes \mu_2), \quad \omega_H(f, g)(x, y) := f(x)g(y),$$

where $\mu_1 \otimes \mu_2$ denotes the product measure (in the sense of measure theory).

We now come back to the setting used in the mathematical description of quantum computing.

Combining two qubits, i.e., two quantum systems with two dimensional state spaces leads, by employing the tensor product to a *2-qubit system* with a 4-dimensional state space

$$\text{span} \{ |0\rangle \otimes |0\rangle, |0\rangle \otimes |1\rangle, |1\rangle \otimes |0\rangle, |1\rangle \otimes |1\rangle \}.$$

Now, an n -qubit system is given by the tensor product of n individual qubits and thus has a 2^n -dimensional state space. For that, we will use the short notation

$$|b_1 b_2 \dots b_n\rangle := |b_1\rangle \otimes |b_2\rangle \otimes \dots \otimes |b_n\rangle$$

and relabel the basis states as $|0\rangle, |1\rangle, \dots, |2^n - 1\rangle$.

A state $|\Phi\rangle$ of an n -qubit system can then be written as superposition of the basis states

$$|\Phi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle$$

with $\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1$.

By definition of our tensor product, we have that $|v_1\rangle \otimes |v_2\rangle$ for all $|v_1\rangle, |v_2\rangle \in H$, or in other words, there is a natural embedding $H \times H \rightarrow H \otimes H$ by means of the mapping ω_H . However, not every state in the tensor product space $H \otimes H$ can be written in a tensor product form.

Definition 8. A state $|\psi\rangle$ is called **product state** if it can be written as

$$|\psi\rangle = |v_1\rangle \otimes \dots \otimes |v_n\rangle, \quad v_i \in H,$$

otherwise it is called **entangled**.

Example 9. (1) The so called Bell state

$$\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$$

is entangled. To see this, consider

$$\begin{aligned} \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle &= (a|0\rangle + b|1\rangle) \otimes (c|0\rangle + d|1\rangle) \\ &= ac|00\rangle + ad|01\rangle + bc|10\rangle + bd|11\rangle, \end{aligned}$$

which implies

$$ac = bd = \frac{1}{\sqrt{2}} \quad \wedge \quad ad = bc = 0,$$

which is a contradiction.

(2) The state

$$\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|10\rangle = \left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \right) \otimes |0\rangle$$

is a product state.

Finally, given two bounded linear operators $T_1, T_2 : H \rightarrow H$, we define the tensor product operator as uniquely defined bounded linear operator satisfying

$$T_1 \otimes T_2 : H \otimes H \rightarrow H \otimes H : v_1 \otimes v_2 \mapsto (T_1 v_1) \otimes (T_2 v_2)$$

and write $T^{\otimes 2} := T \otimes T$.

2.3. Measurement. We now come to the final rule from quantum mechanics needed for quantum computing, which is concerned with the measurement of quantum states. The idea hereby is that we can only observe classical states, which for a qubit are the basic states $|0\rangle, |1\rangle$.

(QM4): A measurement can be employed to a quantum system, which collapses the quantum state to a classical state according to a probability distribution, specified by the Born rule:

Let $|\psi\rangle = \sum_{j=0}^{N-1} \alpha_j |j\rangle$ be the state of a quantum system. Then, the probability of measuring the basic state $|j\rangle$ is $|\alpha_j|^2$.

We note that measuring is a non-reversible process and thus not unitary. It destroys all information hidden in the superposition of a quantum state. Moreover, this shows the non-deterministic nature of quantum mechanics and thus also of quantum computing: Different runs of the same experiment or calculation naturally lead to different outcomes!

A generalization of the process above is given by so called **projective measurement**, which we introduce in the following.

Let H be a state space with $\dim H = N < \infty$ and $P_i, \quad i = 1, \dots, m$ be orthogonal projections onto some subspaces $V_i \subseteq H$ satisfying

$$\sum_{j=1}^m P_j = \text{id}, \quad P_i P_j = \begin{cases} 0 & \text{for } i \neq j \\ P_i & \text{for } i = j \end{cases}.$$

Then, for every $|\Phi\rangle \in H$, we can write

$$|\Phi\rangle = \sum_{j=1}^m |\Phi_j\rangle$$

with $|\Phi_j\rangle = P_j |\Phi\rangle \in V_j$. Orthogonality of the P_j implies orthogonality of the subspaces V_j as well as of the states $|\Phi_j\rangle$. Thus, applying the projector P_j collapses the state $|\Phi\rangle$ to

$|\Phi_j\rangle$ and we have applied some sort of measurement, the projective measurement. The probability of applying P_j or in other words measuring j is

$$\| |\Phi_j\rangle \|^2 = \langle \Phi | P_j P_j | \Phi \rangle = \langle \Phi | P_j | \Phi \rangle$$

and the state collapses to $|\Phi_j\rangle / \| |\Phi_j\rangle \|$. Here, we only have m possible outcomes of measurement. If $m < N = \dim H$, the projective measurement is also called incomplete measurement.

Remark 10. *Note that:*

- In projective measurement, one can choose the spaces V_j and thus the definition of P_j . However, one can not choose which projections P_j will be used in the measurement process! Again, they are selected based on their probabilities.
- If $|\Phi\rangle \in V_j$, measurement will result in j with probability 1, i.e., one will always select P_j .
- Projective measurement is a generalization of the measurement in the computational basis as stated in Born's rule: Selecting $V_j = \text{span}\{|j\rangle\}$, we define P_j as $P_j := |j\rangle\langle j|$ with $\text{rank}(P_j) = 1$ and obtain $|j\rangle = P_j|\phi\rangle$.

Example 11. We will now consider a measurement that, for $|j\rangle$, only decides whether $j < N/2$ or $j \geq N/2$ for $N \in 2\mathbb{N}$. Thus, define the orthogonal projections

$$P_1 := \sum_{j < N/2} |j\rangle\langle j|, \quad P_2 := \sum_{j \geq N/2} |j\rangle\langle j|.$$

Then, for the state

$$|\Phi\rangle = \frac{1}{\sqrt{3}}|1\rangle + \sqrt{\frac{2}{3}}|N\rangle,$$

we measure 1, meaning that the measurement outcome is $|1\rangle = P_1|\Phi\rangle$ in the subspace $V_1 := \text{span}\{|j\rangle : j < N/2\}$ with probability $\|P_1|\Phi\rangle\|^2 = \frac{1}{3}$ or measure 2, meaning that the measurement outcome is $|N\rangle = P_2|\Phi\rangle$ in the subspace $V_2 := \text{span}\{|j\rangle : j \geq N/2\}$ with probability $\|P_2|\Phi\rangle\|^2 = \frac{2}{3}$.

Note that for the state (assuming $N > 4$)

$$|\Phi\rangle = \frac{1}{\sqrt{3}}|1\rangle + \frac{1}{\sqrt{3}}|2\rangle + \frac{1}{\sqrt{3}}|N\rangle,$$

measuring 1 leads to the state $\frac{1}{\sqrt{2}}|1\rangle + \frac{1}{\sqrt{2}}|2\rangle = P_1|\Phi\rangle$, which is again a quantum state.

In the following, we will consider a particular version of projective measurement all the time, which is motivated by the last example. Let a composite quantum system with state space $V \otimes W$ be given and $\{|\phi_1\rangle, \dots, |\phi_n\rangle\} \subset V$, $\{|\psi_1\rangle, \dots, |\psi_m\rangle\} \subset W$ be orthogonal bases. Then, one can define the orthogonal subspaces

$$V_k = \text{span}\{|\phi_k\rangle \otimes |w\rangle : |w\rangle \in W\}$$

and corresponding orthogonal projections $P_k : V \otimes W \rightarrow V_k$. Now, for a state $|\Phi\rangle = \sum_{i,j} \alpha_{i,j} |\phi_i\rangle |\psi_j\rangle$, measuring k in the projective measurement collapses the state (up to normalization) to

$$\sum_j \alpha_{k,j} |\phi_k\rangle |\psi_j\rangle = |\phi_k\rangle \sum_j \alpha_{k,j} |\psi_j\rangle.$$

In other words, only the first state in the composed system collapses to the basis state $|\phi_k\rangle$, while the second state remains a quantum state. This is also called the extended Born rule and allows for measurements of single qubits in composed quantum systems during the calculation process.

3. FUNDAMENTAL OPERATIONS ON QUANTUM COMPUTERS

The currently predominant model of a quantum computer is a quantum circuit model, which is equivalent to the second possible model of a quantum computer, the quantum Turing machine, a modification of the classical Turing machine. Quantum circuits are modifications of classical Boolean circuits, which are finite oriented graphs with AND, OR, and NOT operations

$$\xrightarrow{\text{input}} \{\text{AND, OR, NOT operations}\} \xrightarrow{\text{output}}$$

3.1. Quantum circuits. A quantum circuit diagram is a graphical representation of operations acting on a quantum system composed of finitely many qubits. Manipulations are usually done on a small number of qubits.

Definition 12. A unitary operator U acting on a small number of qubits is called *gate*.

A gate is similar to e.g., AND, XOR, and OR gates in classical computing and some useful examples are given in the next subsection.

The assumptions for a quantum circuit are as follows:

- A quantum circuit is a directed graph (emulating time evolution), where input is written on the left and output on the right.
- Each qubit is represented by one wire.
- All gates have the same number of input and output wires (as they have to be reversible). By the tensor product structure of combined quantum systems, one has that the tensor product operators $\text{id} \otimes U$, $U \otimes \text{id}$ are unitary, provided U is unitary. Thus, the application of a gate to only parts of a quantum circuit is well defined. Simple gates are usually written using the box notation $\boxed{\cdot}$.
- Measurement of a state is an allowed operation and denoted by $\boxed{\text{meter}}$.
- Classical (deterministic!) states can appear in quantum circuits and are denoted by double wires.

Example 13. A quantum circuit with one qubit and unitary operation is given in the following figure.

$$|0\rangle \text{ --- } \boxed{U} \text{ --- } U|0\rangle$$

FIGURE 1. Quantum circuit for application of a unitary U .

A quantum circuit with three qubits and an unitary operation only acting on the first two qubits is depicted in the following.

$$\begin{array}{c} |0\rangle \text{ --- } \\ |1\rangle \text{ --- } \\ |0\rangle \text{ --- } \end{array} \left. \begin{array}{c} \boxed{U} \\ \end{array} \right\} \begin{array}{c} \text{---} \\ \text{---} \\ |0\rangle \end{array}$$

FIGURE 2. Circuit for application of a unitary U on two qubits only.

3.2. Single qubit gates. We now introduce some commonly used gates, starting with gates acting on only one qubit. As these gates have to be unitary operators from two dimensional Hilbert spaces on itself, we can represent them as 2×2 unitary matrices.

The **bit flip** gate \boxed{X} (*NOT*) swaps $|0\rangle$ and $|1\rangle$ and is defined by

$$\boxed{X} := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

The **phase flip** gate \boxed{Z} only mirrors $|1\rangle$ and is defined by

$$\boxed{Z} := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Note that this is a special case of the **phase-gate** $\boxed{R_\Phi}$, which rotates $|1\rangle$ by $\Phi \in [-\pi, \pi]$, defined as

$$\boxed{R_\Phi} := \begin{pmatrix} 1 & 0 \\ 0 & e^{i\Phi} \end{pmatrix}$$

$R_{\pi/4}$ is also called **T-gate**.

The **Hadamard-gate** \boxed{H} is the most important single qubit operator. It is defined as

$$\boxed{H} := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

Since

$$\boxed{H} |0\rangle = \boxed{H} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \simeq \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle,$$

we see that the probability for $|0\rangle$ and $|1\rangle$ is equal. Furthermore, we have

$$\boxed{H} = \boxed{H}^T = \boxed{H}^{-1} \implies \boxed{H} \left(\frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle \right) = |0\rangle.$$

From now on, we will only use the box notation in quantum circuits and write H instead of \boxed{H} in formulas.

3.3. Two qubit gates. We now consider gates acting on two qubits, which thus can be represented by 4×4 -matrices. The main idea for two and subsequently three qubit gates is to implement classical logic operations such as AND, OR, XOR.

We start with the logic operation XOR, i.e., given $a, b \in \{0, 1\}$, we want to implement the operation $(a, b) \mapsto a \text{ XOR } b$ defined by the following truth table:

	b	
a \	0	1
0	0	1
1	1	0

However, this operation is mapping $\{0, 1\}^2 \rightarrow \{0, 1\}$ and thus can not be reversible and unitary. In order to fix this, we define the mapping $\{0, 1\}^2 \rightarrow \{0, 1\}^2 : (a, b) \mapsto (a, a \text{ XOR } b)$ as

$$\frac{(a, b)}{(a, a \text{ XOR } b)} \mid \begin{array}{cccc} (0, 0) & (0, 1) & (1, 0) & (1, 1) \\ (0, 0) & (0, 1) & (1, 1) & (1, 0) \end{array}$$

This is a permutation and thus unitary, which can be represented by the unitary matrix

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \quad (2)$$

This operation is called **Controlled Not** (CNOT) and is visualized in the circuit diagram as shown in the circuit diagram below.

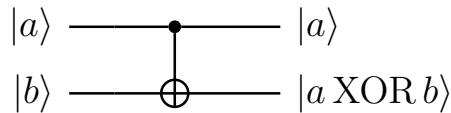


FIGURE 3. Controlled Not (CNOT) circuit diagram.

Note that we have

$$\begin{aligned} \text{CNOT}|0\rangle|b\rangle &= |0\rangle|b\rangle \\ \text{CNOT}|1\rangle|b\rangle &= |1\rangle|1-b\rangle, \end{aligned}$$

i.e., the first qubit is unchanged under CNOT (the so called *control qubit*), but the value of the first qubit influences the output of the second qubit (the so called *target qubit*), thus the name of a *controlled* operation.

Whenever a qubit controls another qubit-operation it is noted with a dot and vertical line in the circuit diagram and the gate applied to the controlled qubit is denoted by \oplus . In this sense, we will also use the short notation

$$|x\rangle \oplus |y\rangle := \text{CNOT}|x\rangle|y\rangle.$$

Example 14. *The combination of a Hadamard and a CNOT gate can be used to define the bell state, since*

$$\text{CNOT}(H(|0\rangle)|0\rangle) = \text{CNOT}\left(\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|10\rangle\right) = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle.$$

Remark 15. *The CNOT gate together with the Hadamard gate H and the phase gate T form a universal set of gates in the sense of approximation, meaning that any unitary operator U can be approximated well by a circuit consisting of a combination of these three gates.*

3.4. Three qubit gates. We now try to realize the AND operation, i.e., $(a, b) \mapsto a \text{ AND } b$ with $a, b \in \{0, 1\}$, defined by the following truth table:

		b	
	a	0	1
0		0	0
1		0	1

Again, this is not a unitary operation. However, the mapping $\{0, 1\}^2 \rightarrow \{0, 1\}^2 : (a, b) \mapsto (a, a \text{ AND } b)$ is not bijective and thus not unitary as well.

The solution to this problem is to introduce an extra qubit $|c\rangle$, and define the mapping $\{0, 1\}^3 \rightarrow \{0, 1\}^3 : (a, b, c) \mapsto (a, b, c \text{ XOR}(a \text{ AND } b))$ which reads as

(a, b, c)	$(0, 0, 0)$	$(0, 0, 1)$	$(0, 1, 0)$	$(0, 1, 1)$	$(1, 0, 0)$	$(1, 0, 1)$	$(1, 1, 0)$	$(1, 1, 1)$
	$(0, 0, 0)$	$(0, 0, 1)$	$(0, 1, 0)$	$(0, 1, 1)$	$(1, 0, 0)$	$(1, 0, 1)$	$(1, 1, 1)$	$(1, 1, 0)$

This operation (acting on qubits) is called **Toffoli gate** (or CCNOT gate) and is visualized in the circuit diagram shown below.

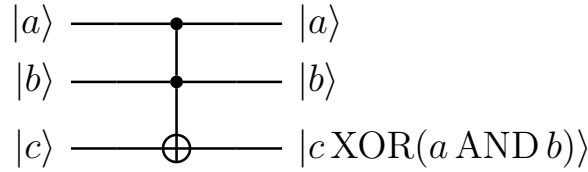


FIGURE 4. Toffoli gate circuit diagram.

Thus we see that

$$|c\rangle = 0 \quad \implies \quad \text{output state } |a\rangle|b\rangle|a \text{ AND } b\rangle$$

The extra qubit $|c\rangle$ is called *ancilla*-qubit as it is only used to steer the wanted output (by feeding it with the input $|0\rangle$). Moreover, the first two qubits are left unchanged, but their values directly influence the output in the third qubit, which explains the name CCNOT gate (as controlled controlled NOT).

The previous two defined gates directly motivate the following more general definition.

Definition 16. Let U be a unitary n -qubit operation and let $\text{id} \in \mathbb{C}^{2^n \times 2^n}$, then the operation

$$\begin{pmatrix} \text{id} & 0 \\ 0 & U \end{pmatrix} \in \mathbb{C}^{2^{n+1} \times 2^{n+1}}$$

is called **controlled-U**.

Remark 17. The Toffoli gate can be used to implement the logic operation NOT: Fix the value of the first two qubits, then

$$\text{Toffoli}(|1\rangle|1\rangle|a\rangle) = |1\rangle|1\rangle|1 - a\rangle.$$

Since

$$a \text{ OR } b = \text{NOT}(\text{NOT } a \text{ AND NOT } b)$$

Toffoli gates can realize all the logic operations AND, NOT, OR and thus any classical circuit can be implemented by a circuit of Toffoli gates.

3.5. **Quantum information processing.** We now continue with some basics of quantum information processing.

So far, we know that:

- 2 states can only be distinguished with probability 1, if they are in orthogonal subspaces.
- There are three possible manipulations of states $|\psi\rangle$:
 - **(Ancilla)** combining a known state $|A\rangle$ with $|\psi\rangle$ we obtain $|\psi\rangle|A\rangle$, enlarging the dimension of the state space.
 - **(Unitary operation)** Calculate $U|\psi\rangle$ with U unitary.
 - **(Measurement)** Collapse the state $|\psi\rangle$ to some basis state $|j\rangle$.

A natural question in information processing is whether we can also copy or save information? Unfortunately, the answer is *no*, as stated by the *no-cloning theorem*.

3.5.1. *The no-cloning theorem.* Let H be a state space for 2 quantum systems

- A that contains the state to be copied $|\psi\rangle$,
- B that contains the target of the copy, initialized as $|0\rangle$

and one quantum system

M the copy-machine with input state $|M_0\rangle$.

The goal is to find an operation

$$|\psi\rangle|0\rangle|M_0\rangle \rightarrow |\psi\rangle|\psi\rangle|M_\psi\rangle \quad (3)$$

that works for all states $|\psi\rangle$ in A .

Theorem 18 (No-cloning theorem). *Suppose $S \subseteq H$ such that S contains at least a pair of linearly independent, non-orthogonal states. Then,*

\nexists unitary operator U on S , that can copy all states in S in the sense of (3).

Proof. Let $|\xi\rangle, |\eta\rangle \in S$ be different and non-orthogonal states. Assume there exists a unitary operator U with

$$\begin{aligned} U|\xi\rangle|0\rangle|M_0\rangle &= |\xi\rangle|\xi\rangle|M_\xi\rangle \\ U|\eta\rangle|0\rangle|M_0\rangle &= |\eta\rangle|\eta\rangle|M_\eta\rangle. \end{aligned}$$

Since U is unitary, U preserves the inner product of both input states, i.e.,

$$\begin{aligned} \langle\xi|\eta\rangle\langle\xi|\eta\rangle\langle M_\xi|M_\eta\rangle &= \langle\xi|\xi\rangle\langle\xi|M_\xi\rangle, \langle\eta|\eta\rangle\langle\eta|M_\eta\rangle \\ &= \langle\xi|0\rangle\langle 0|M_0\rangle, \langle\eta|0\rangle\langle 0|M_0\rangle = \langle\xi|\eta\rangle \underbrace{\langle 0|0\rangle}_{|\cdot|=1} \underbrace{\langle M_0|M_0\rangle}_{|\cdot|=1}. \end{aligned}$$

Hence, we have

$$|\langle\xi|\eta\rangle|^2 |\langle M_\xi|M_\eta\rangle| = |\langle\xi|\eta\rangle|,$$

and since $|\langle\xi|\eta\rangle| \neq 0$ as $|\xi\rangle, |\eta\rangle$ are non-orthogonal, we get

$$|\langle\xi|\eta\rangle| |\langle M_\xi|M_\eta\rangle| = 1.$$

Using the Cauchy-Schwarz inequality, we also know that

$$|\langle M_\xi|M_\eta\rangle| \leq \underbrace{|\langle M_\xi|M_\xi\rangle|}_{=1} \underbrace{|\langle M_\eta|M_\eta\rangle|}_{=1} = 1.$$

Combining the last two equations leads to

$$1 \leq |\langle\xi|\eta\rangle| \leq \|\xi\| \|\eta\| = 1.$$

However, this is a contradiction as $|\langle \xi | \eta \rangle| = 1$ for unit vectors implies $|\xi\rangle = e^{i\Phi}|\eta\rangle$ for some $\Phi \in [-\pi, \pi)$ and thus linear dependency. \square

As a converse to the no-cloning theorem, in quantum information processing it is also impossible to delete a state.

Corollary 19 (No-deleting theorem). *Assume the same setting as in Theorem 18. Then, there exists no unitary operator U such that*

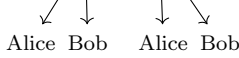
$$U : |\psi\rangle|\psi\rangle|M_0\rangle \rightarrow |\psi\rangle|0\rangle|M_\psi\rangle.$$

However, there is a solution for information transfer in exploiting *entanglement*.

3.5.2. *Quantum teleportation*. We now show on the example of quantum teleportation how entanglement of quantum states can be used for information transport.

Consider the following setting: Alice and Bob are far away from each other but want to share some information:

- Alice has a qubit $|\alpha\rangle := \alpha_0|0\rangle + \alpha_1|1\rangle$, Bob wants information about Alice's qubit.
- Alice and Bob share an entangled state in a combined quantum system

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle).$$


Alice Bob Alice Bob

How can Bob get information about Alice's qubit without physically transferring Alice's qubit?

The answer to this question is given by so called *quantum teleportation*, which is composed of the following steps:

Step 1: There are 3 qubits involved:

- 1. qubit: Alice's qubit $|\alpha\rangle$,
- 2. qubit: Alice's part of the entangled qubit, and
- 3. qubit: Bob's part of the entangled qubit.

These can be combined to a quantum system by

$$(\alpha_0|0\rangle + \alpha_1|1\rangle) \otimes \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle).$$

Step 2: Alice applies CNOT to the first and second qubit, i.e.,

$$\frac{\alpha_0}{\sqrt{2}}|0\rangle(|00\rangle + |11\rangle) + \frac{\alpha_1}{\sqrt{2}}|1\rangle(|10\rangle + |01\rangle).$$

Step 3: Alice applies the Hadamard gate H to the first qubit to obtain

$$\begin{aligned} & \frac{1}{2}\alpha_0(|0\rangle + |1\rangle)(|00\rangle + |11\rangle) + \frac{1}{2}\alpha_1(|0\rangle - |1\rangle)(|10\rangle + |01\rangle) \\ &= \frac{1}{2}|00\rangle(\alpha_0|0\rangle + \alpha_1|1\rangle) + \frac{1}{2}|01\rangle(\alpha_0|1\rangle + \alpha_1|0\rangle) \\ & \quad + \frac{1}{2}|10\rangle(\alpha_0|0\rangle - \alpha_1|1\rangle) + \frac{1}{2}|11\rangle(\alpha_0|1\rangle - \alpha_1|0\rangle). \end{aligned}$$

Step 4: Alice measures both of her qubits. The states collapses to one of 4 possible states with probability $P = 1/4$.

measurement	state after measurement
$ 00\rangle$	$ 00\rangle \alpha\rangle$
$ 01\rangle$	$ 01\rangle X \alpha\rangle$ (bitflip)
$ 10\rangle$	$ 10\rangle Z \alpha\rangle$ (phaseflip)
$ 11\rangle$	$ 11\rangle XZ \alpha\rangle$ (bit+phaseflip)

Step 5: Alice sends the result of the measurement (e.g. 10) of her two qubits to Bob on a classical channel.

Step 6: Bob looks up the corresponding operation in the table above and applies the inverse transformation to his qubit. In this case (10), Bob applies the inverse of Z (which is Z itself) to his qubit. Thus Bob's qubit is guaranteed to be in the state $|\alpha\rangle$, which was the state of Alice initial qubit.

The quantum circuit for quantum teleportation is depicted in the following.

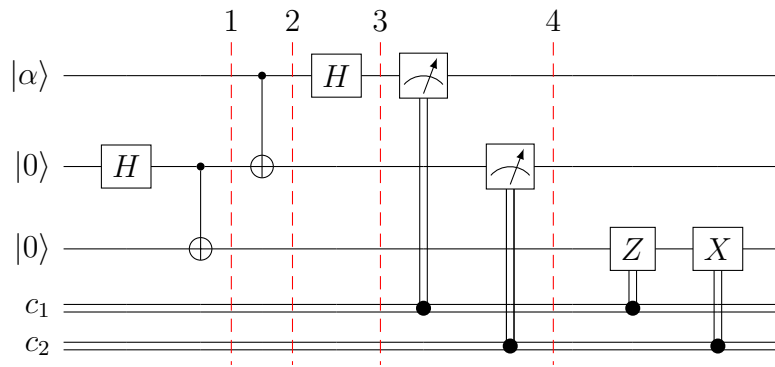


FIGURE 5. Quantum circuit for quantum teleportation.

Remark 20. *Quantum teleportation warrants the following remarks:*

- *Entanglement is a non-local interaction: physical processes in space between Alice and Bob have no influence on the information transfer.*
- *Only information can be transported, not matter. No beaming as in Star Trek!*
- *Alice has to measure her qubits. This is invasive and the state of her qubit collapses (it is no longer in superposition!).*

3.6. Encoding. It is not immediately clear how to put a number, a vector or a matrix into a quantum computer. Here, we present some possible encodings for that.

Binary encoding: Let $n \in \mathbb{N}$ and $x \in \mathbb{N}$ with $x < 2^n$. Then, x can be written as $x = \sum_{j=0}^{n-1} \alpha_j 2^j$ with $\alpha_j \in \{0, 1\}$. In the same way, certain rational numbers can be represented for some $0 \leq m \leq n$ by

$$\frac{x}{2^m} = \sum_{j=0}^{n-1} \alpha_j 2^{j-m}.$$

Thus, we can encode a number x of this form by a sequence of qubits as

$$|x\rangle = |\alpha_{n-1}\rangle |\alpha_{n-2}\rangle \dots |\alpha_0\rangle.$$

This type of encoding is also called basis encoding, as the information in x is directly encoded in the computational basis.

Not every rational can be written exactly in this form, but for fixed m one obtains at least a good approximation (of order 2^{-m}) with this encoding.

A drawback is that one needs quite a lot of qubits for this type of encoding.

Example 21. *The number $x = 14$ has the binary representation $x = 0 \cdot 1 + 1 \cdot 2 + 1 \cdot 2^2 + 1 \cdot 2^3$ and thus is encoded in the 4-qubit register $|1110\rangle$.*

Amplitude encoding: An alternative to basis encoding is to encode information in the amplitude of a quantum state. Given a vector $x \in \mathbb{R}^N$, define the quantum state

$$|\psi_x\rangle = \frac{1}{\gamma} \sum_{j=1}^N x_j |j-1\rangle, \quad \gamma^2 = \sum_{j=1}^N |x_j|^2.$$

Note that the normalization γ is necessary to obtain a quantum state.

Amplitude encoding is particularly useful for vectors, as one only needs $n \sim \log N$ qubits to encode a vector in \mathbb{R}^N .

However, a drawback is that manipulations of the induced quantum state have to operate on the amplitudes and final results are only obtained after possibly many measurements, which might be inefficient.

Example 22. *The vector $(1, 4, 2, 2)^T \in \mathbb{R}^4$ has norm $\gamma = \sqrt{1 + 16 + 4 + 4} = 5$ and can be encoded as the quantum state*

$$\begin{aligned} |\psi_x\rangle &= \frac{1}{5}|0\rangle + \frac{4}{5}|1\rangle + \frac{2}{5}|2\rangle + \frac{2}{5}|3\rangle \\ &= \frac{1}{5}|00\rangle + \frac{4}{5}|01\rangle + \frac{2}{5}|10\rangle + \frac{2}{5}|11\rangle \end{aligned}$$

and a 2 qubit register is enough to store the information in x .

Note that, if a vector does not have length 2^n for some $n \in \mathbb{N}$, one can fill it up with zeros to achieve that.

Block encoding: We now try to encode a matrix A in a quantum computer. This is not encoded as a data structure, but rather as application of the matrix by means of matrix-vector multiplication.

For unitary matrices this can be (approximately) realized as a combination of elementary gates as previously discussed. However, most matrices are not unitary. In order to deal with that, we apply the same idea we already employed when realizing basic logical operations of embedding into a higher dimensional space, i.e., we want to realize a unitary U_A of the form

$$U_A = \frac{1}{\gamma} \begin{pmatrix} A & \star \\ \star & \star \end{pmatrix}.$$

Then, A can be obtained by

$$A = \gamma(\langle 0| \otimes I)U_A(|0\rangle \otimes I).$$

Matrix-vector multiplication with A is thus written as

$$U_A|0b\rangle = |0\rangle \frac{1}{\gamma} A|b\rangle + |1\rangle |\psi\rangle$$

for some state $|\psi\rangle$. Thus, measuring the first qubit (register) and only keeping it, if one obtained 0, realizes the matrix vector product $A|b\rangle$.

The precise definition of a block encoding is given in the following.

Definition 23. Let $A \in \mathbb{C}^{N \times N}$ where $N = 2^n$. Then, we call a unitary matrix $U_A \in \mathbb{C}^{(N+M) \times (N+M)}$ with $M = 2^m$ a (γ, m, ε) -block encoding of A , if

$$\|A - \gamma(\langle 0^m | \otimes I_n)U_A(|0^m\rangle \otimes I_n)\| \leq \varepsilon.$$

The circuit for a general block encoding of A reads as.

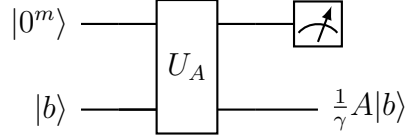


FIGURE 6. Quantum circuit for block encoding of A .

Example 24. The matrix $A = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$ is not unitary. However, its $(2, 1, 0)$ block encoding

$$U = \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & -1 \\ 1 & 1 & -1 & 1 \\ 1 & -1 & 1 & 1 \\ -1 & 1 & 1 & 1 \end{pmatrix}$$

is unitary and, up to the scaling factor $1/2$, the first 2×2 -block is A .

Block encodings also allow for arithmetical operations like addition, multiplication or tensor products.

3.7. Complexity. The complexity of a quantum circuit can be measured in various ways.

A straight forward way would be counting the number of gates applied to qubits, which is called **gate complexity**, which can be interpreted as serial time complexity. A quantum algorithm is considered gate efficient, if the gate complexity is $\mathcal{O}(\text{poly}(n))$, where n is the number of qubits needed to represent the input and $\text{poly}(n)$ denotes polynomial dependence on n .

A second measure would be the **depth of the circuit**, i.e., the maximal number of gates along any path from input to output, which can be interpreted as parallel time complexity. Circuit depth gives an estimate on how long quantum states have to be preserved. With the current noise affected hardware, this is important for concrete calculations on quantum computers.

The third way to measure the complexity of a quantum circuit is in terms of **query complexity**, which counts how often we use a so called *oracle*, which is a circuit or a data structure that can be used as a black box in the algorithm (e.g. evaluation of a certain function, more on that later).

Query complexity hides the effort needed to implement the oracle, which can be rather significant on a quantum computer. If one has a concrete quantum circuit for the oracle, one immediately can deduce the gate complexity from the query complexity.

4. CLASSICAL QUANTUM ALGORITHMS

In this section, we present some of the first algorithms that demonstrated the actual usefulness of a device that does computations on qubits by laws of quantum mechanics.

In a nutshell, a quantum algorithm is an algorithm that harnesses the inherent power of qubits to do many calculations at once. The problem in quantum computing is, however, that the result of such calculations is a quantum state and thus in order to obtain a deterministic result, one has to take a measurement. As the same experiment can produce different outcomes, one usually has to retake the calculations multiple times to obtain a certain result with very high probabilities. This process in turn can be costly and may offset any advantages in computational effort gained by using qubits.

Thus, the main objective of a quantum algorithm is to make clever manipulations of quantum states (in physics called destructive interference) such that with few (projective) measurements a useful result is obtained with high probability. Hereby, the term **quantum advantage** means that a quantum algorithm can solve a given problem on a quantum computer with significant speed-up (in the best case exponential speed-up) compared to a classical computer.

4.1. Deutsch-Josza-Algorithm. We start with the arguably first quantum algorithm that provided a possible exponential speed-up, discovered by David Deutsch and Richard Josza in 1992.

Problem formulation. Let $\mathcal{F} : \{0, 1\}^n \rightarrow \{0, 1\}$ be a function, which satisfies either

- \mathcal{F} is constant, i.e., $\mathcal{F} \equiv 0$ or $\mathcal{F} \equiv 1$, or
- \mathcal{F} is *balanced*, i.e., exactly half of the inputs (x_1, \dots, x_n) lead to $\mathcal{F}(x_1, \dots, x_n) = 1$ and the other half leads to $\mathcal{F}(x_1, \dots, x_n) = 0$.

The goal is to determine, which of the two cases holds for a fixed given \mathcal{F} .

While the practical relevance of this problem is very limited, it is a instructive toy problem to demonstrate quantum advantage.

Solving the problem on classical computers. To exclude the *balanced* case, on a classical computer, one needs to check at least $2^{n-1} + 1$ inputs $(x_1, \dots, x_n) \in \{0, 1\}^n$. Thus, with the problem size of $N = 2^n$, we need exactly $N/2 + 1$ function evaluations of \mathcal{F} .

Physical inspiration for the quantum algorithm. Consider the setup depicted in fig. 7: A light source (e.g. the sun) emits light rays which are directed towards a wall with 2^n holes $(x_1, \dots, x_n) \in \{0, 1\}^n$. The light passes through the holes and is detected by a detector. Assume that the holes are arranged such that the light amplitude at the detector $\simeq (-1)^{\mathcal{F}(y)}$ if the light travels through hole y . The light intensity at the detector is determined by the phase difference of light paths (interference) and is given by

$$\simeq \sum_{y \in \{0, 1\}^n} (-1)^{\mathcal{F}(y)} = \begin{cases} 0 & \mathcal{F} \text{ is balanced} \\ \pm 1 & \mathcal{F} \text{ is constant} \end{cases} . \quad (4)$$

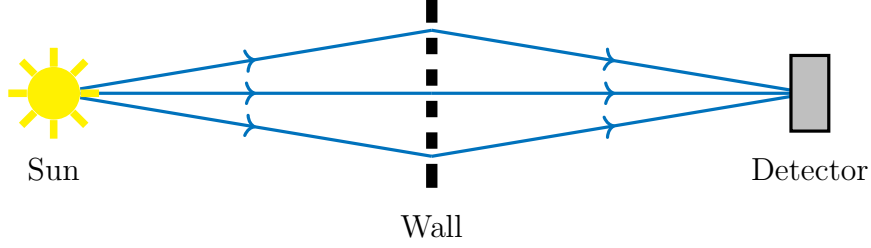


FIGURE 7. Physical inspiration for Deutsch-Josza algorithm.

Quantum algorithm. We may identify $(x_1, \dots, x_n) \in \{0, 1\}^n$ with a bit string and thus with the basis element $|i\rangle$, where $i = x_1 + x_2 2 + x_3 2^2 + \dots + x_n 2^{n-1}$. In this sense, we define $\mathcal{F}(|i\rangle) := |\mathcal{F}(x_1, \dots, x_n)\rangle$. An essential part of this (and many other algorithms) is a *query* or so called *oracle* access to the function \mathcal{F} : Given a 1-qubit state $|b\rangle$ and $\mathcal{F}(x_1, \dots, x_n) \in \{0, 1\}$, define

$$\mathcal{Q}_{\mathcal{F}}(|i\rangle \otimes |b\rangle) = |i\rangle \otimes (|b\rangle \oplus \mathcal{F}(|i\rangle)). \quad (5)$$

In particular, for $|b\rangle = |-\rangle := \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$, we have

$$\begin{aligned} \mathcal{Q}_{\mathcal{F}}(|i-\rangle) &= |i\rangle \otimes \frac{1}{\sqrt{2}} [|0\rangle \oplus \mathcal{F}(|i\rangle) - |1\rangle \oplus \mathcal{F}(|i\rangle)] \\ &= \begin{cases} |i\rangle \otimes \frac{1}{\sqrt{2}} [|0\rangle - |1\rangle] & \text{if } \mathcal{F}(|i\rangle) = |0\rangle \\ |i\rangle \otimes \frac{1}{\sqrt{2}} [|1\rangle - |0\rangle] & \text{if } \mathcal{F}(|i\rangle) = |1\rangle \end{cases} \\ &= (-1)^{\mathcal{F}(|i\rangle)} |i-\rangle. \end{aligned}$$

Note that we identify $\mathcal{F}(|i\rangle)$ with $\mathcal{F}(x_1, \dots, x_n)$ instead of $|\mathcal{F}(x_1, \dots, x_n)\rangle$, i.e., the last equality only makes sense for basis elements $|i\rangle$. Furthermore, $\mathcal{Q}_{\mathcal{F}}$ is unitary since

$$|b\rangle \mapsto |b\rangle \oplus \mathcal{F}(|a\rangle)$$

is just a permutation.

We make extensive use of the Hadamard gate H , for which we recall

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \quad H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

The following lemma, which follows from direct calculation provides formulas for multiple application of the Hadamard gate.

Lemma 25. *Let $N = 2^n$ with $n \in \mathbb{N}$. There holds*

$$\begin{aligned} H^{\otimes n} |0^n\rangle &:= \bigotimes_{i=1}^n H|0\rangle = \bigotimes_{i=1}^n \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) = \sum_{(x_1, \dots, x_n) \in \{0, 1\}^n} \frac{1}{\sqrt{2^n}} |x_1 \dots x_n\rangle \\ &= \frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} |i\rangle. \end{aligned}$$

Moreover, for $i = x_1 + 2x_2 + \dots + 2^{n-1}x_n$, we have

$$\begin{aligned} H^{\otimes n}|i\rangle &:= \bigotimes_{i=1}^n H|x_i\rangle = \bigotimes_{i=1}^n \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{x_i}|1\rangle) \\ &= \sum_{(y_1, \dots, y_n) \in \{0,1\}^n} \frac{1}{\sqrt{N}} (-1)^{x \cdot y} |y_1 \dots y_n\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{2^n-1} (-1)^{i \cdot j} |j\rangle, \end{aligned}$$

where $i \cdot j := x \cdot y := \sum_{k=1}^n x_k y_k$.

With the use of the lemma, we can state the Deutsch-Josza algorithm.

Algorithm 1 (Deutsch-Josza Quantum Algorithm)

Input: Initial State $|0^n\rangle := |0\rangle \otimes \dots \otimes |0\rangle$

1: Apply $\boxed{\text{H}}$ to each qubit to obtain uniform state $\frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} |i\rangle$.

2: Tensorize result with $|-\rangle$ to obtain $\frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} |i-\rangle$.

3: Apply query with $\boxed{\text{Q}_{\mathcal{F}}}$ to obtain $\frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} (-1)^{\mathcal{F}(i)} |i-\rangle$.

4: Ignore last qubit $|-\rangle$ and apply $\boxed{\text{H}}$ to remaining state to obtain with Lemma 25

$$\frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} \sum_{j=0}^{2^n-1} (-1)^{i \cdot j} (-1)^{\mathcal{F}(i)} |j\rangle.$$

5: Measurement of the remaining n -qubit register. The amplitude of the $j = 0$ -state (which is $|0^n\rangle$) is $\frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} (-1)^{i \cdot 0} (-1)^{\mathcal{F}(i)}$.

Now, the information whether \mathcal{F} is balanced or constant is encoded in the result of the algorithm since:

- \mathcal{F} is *balanced* $\implies \frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} (-1)^{\mathcal{F}(i)} = 0$
- $\mathcal{F} \equiv 1$ $\implies \frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} (-1)^{\mathcal{F}(i)} = -1$
- $\mathcal{F} \equiv 0$ $\implies \frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} (-1)^{\mathcal{F}(i)} = 1$

Thus, if the result of our measurement of the first n -qubit register is the state $|0^n\rangle$, we know *for sure* that \mathcal{F} is balanced. If the result of the measurement is any other state, we know that \mathcal{F} is constant.

Complexity of the quantum algorithm. Algorithm 3 needs $\mathcal{O}(1)$ steps, $\mathcal{O}(\log_2(N))$ quantum gates, $\mathcal{O}(\log_2(N))$ qubits, and one query of the oracle $Q_{\mathcal{F}}$. Thus, the quantum algorithm offers an *exponential speedup* to the classical algorithm.

Example 26. We consider the function

$$\mathcal{F}(x_1, x_2, x_3) := x_1 + x_2x_3 \bmod 2$$

which is balanced since $\mathcal{F}(0, x_2, x_3) = 1 - \mathcal{F}(1, x_2, x_3)$. For the algorithm to work, \mathcal{F} needs to be implemented in a quantum circuit which we can realize as

$$\mathcal{F}(x_1, x_2, x_3) = x_1 \text{ XOR}(x_2 \text{ AND } x_3).$$

The circuit diagram looks as follows.

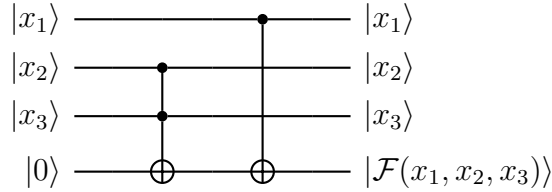


FIGURE 8. Quantum circuit for \mathcal{F} , the last qubit is an ancilla qubit.

Next, we implement the Query $Q_{\mathcal{F}}(|i-\rangle) = |i\rangle \otimes (|i-\rangle \oplus \mathcal{F}(|i\rangle))$, where the state $|-\rangle$ is given by $HX|0\rangle$.

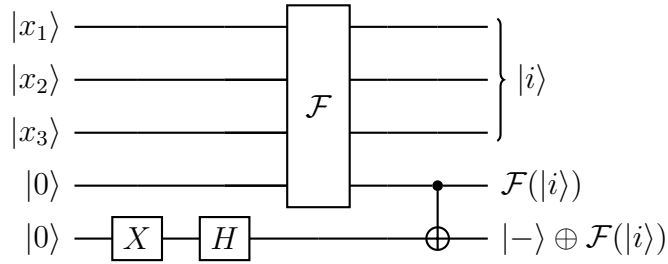


FIGURE 9. Implementation of the query $Q_{\mathcal{F}}$.

A full implementation of the Deutsch-Josza algorithm as a quantum circuit can be seen in Figure 10. Note that in the implementation, we actually applied \mathcal{F}^{-1} to the fourth qubit, which was not part of Algorithm 3. This is needed due to entanglement, see Remark 27 below.

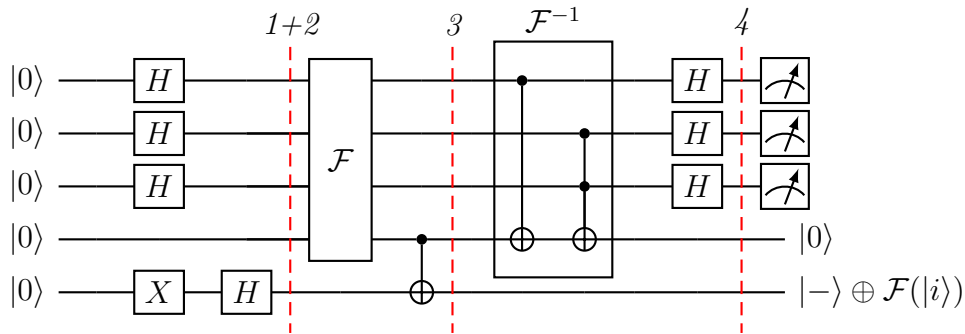


FIGURE 10. Deutsch-Josza algorithm circuit diagram.

We note that there are many ways to implement the query $\boxed{Q_{\mathcal{F}}}$ and the previous circuit is not the most efficient implementation. An alternative using the controlled Z-gate (recall $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$) is depicted in the following.

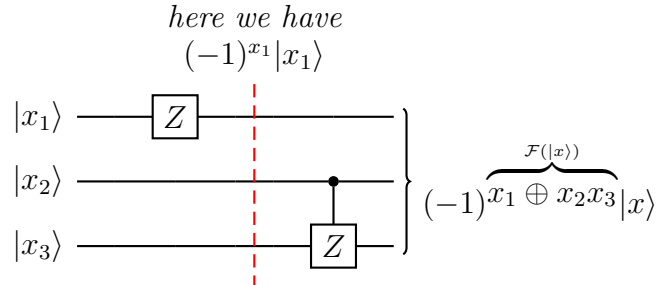


FIGURE 11. Quantum circuit for $\boxed{Q_{\mathcal{F}}}$ using controlled Z.

The controlled Z gate is only active if $x_2 = 1$ and thus realizes $(-1)^{x_2 \cdot x_3}|x\rangle$. Consequently the quantum circuit realizes \mathcal{F} in the exponent of (-1) .

Remark 27. Why do you have to uncompute \mathcal{F} in order to remove entanglement? After algorithm 1, we have

$$\frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} (-1)^{\mathcal{F}(|i\rangle)} |i\rangle$$

but in the implementation, we actually have

$$\frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} (-1)^{\mathcal{F}(|i\rangle)} |i\rangle \otimes |a_i\rangle \otimes |-\rangle$$

for some ancilla qubit $|a_i\rangle$. We may ignore the last qubit since it is not entangled. Algorithm 1 applies $H^{\otimes n}$ to obtain

$$\frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (-1)^{\mathcal{F}(|i\rangle)} (-1)^{i \cdot j} |j\rangle \otimes |a_i\rangle.$$

Now, partial measurement of this state might produce $|0^n\rangle$ (summand $j = 0$) in the first register, which gives the quantum state

$$\frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} (-1)^{\mathcal{F}(|i\rangle)} |0\rangle \otimes |a_i\rangle.$$

However, from this, we can not deduce that a balanced \mathcal{F} implies that the amplitude has to be zero (as in the summation over i the additional tensorized state $|a_i\rangle$ comes into play).

Quantum advantage? As seen in the complexity above, in principle, the Deutsch-Josza algorithm was the first quantum algorithm with theoretical exponential speed-up over any deterministic algorithm on a classical computer.

The word deterministic is key here: One can actually consider a *randomized* algorithm, with output denoted by $\mathcal{R}(\mathcal{F})$, on a classical computer as follows:

- 1: Generate two random vectors $x, y \in \{0, 1\}^n$.
- 2: Evaluate $\mathcal{F}(x), \mathcal{F}(y)$.
- 3: Output:

$$\begin{aligned} \mathcal{R}(\mathcal{F}) &= 1 && \text{if } \mathcal{F}(x) = \mathcal{F}(y) = 1 \\ \mathcal{R}(\mathcal{F}) &= 0 && \text{if } \mathcal{F}(x) = \mathcal{F}(y) = 0 \\ \mathcal{R}(\mathcal{F}) & \text{balanced} && \text{if } \mathcal{F}(x) \neq \mathcal{F}(y). \end{aligned}$$

This algorithm is correct, if \mathcal{F} is constant and answers correctly with probability $1/2$ if \mathcal{F} is *balanced*.

Apply the algorithm k -times with i.i.d. random samples to obtain outputs $(\mathcal{R}_1(\mathcal{F}), \dots, \mathcal{R}_k(\mathcal{F}))$.

$$\text{Return} \begin{cases} 1 & \text{if } \mathcal{R}_1(\mathcal{F}) = \dots = \mathcal{R}_k(\mathcal{F}) = 1 \\ 0 & \text{if } \mathcal{R}_1(\mathcal{F}) = \dots = \mathcal{R}_k(\mathcal{F}) = 0 \\ \text{balanced} & \text{otherwise.} \end{cases}$$

This results in an algorithm with error probability of 2^{-k} and cost $\mathcal{O}(k)$.

4.2. Simon's algorithm. While the Deutsch-Josza algorithm provides quantum advantage over a deterministic algorithm, it does not over a randomized classical algorithm.

In the following, we consider the historically first algorithm to achieve a theoretical quantum advantage over randomized classical algorithms as well.

Simon's problem. Given $i, j \in \{0, 1\}^n$, we define $i \oplus j := (i_1 \oplus j_1, \dots, i_n \oplus j_n)$. Let a function $\mathcal{F} : \{0, 1\}^n \rightarrow \{0, 1\}^n$ be given such that there exists $s \in \{0, 1\}^n$ with

$$\mathcal{F}(x) = \mathcal{F}(y) \iff x = y \text{ or } x \oplus s = y \quad \forall x, y \in \{0, 1\}^n.$$

The goal is to determine s .

We note that if $s = \{0\}^n$, Simon's problem translates to $\mathcal{F}(x) = \mathcal{F}(y) \iff x = y$, i.e., \mathcal{F} has to be injective.

If $s \neq \{0\}^n$ the assumption on \mathcal{F} states that it is not injective, but a two-to-one function.

Example 28. Let $\mathcal{F} : \{0, 1\}^2 \rightarrow \{0, 1\}^2$ be given by

$$\begin{aligned} f(0, 0) &= (0, 0), & f(1, 0) &= (1, 1) \\ f(0, 1) &= (0, 0), & f(1, 1) &= (1, 1). \end{aligned}$$

Then, as $(0, 0) \oplus (0, 1) = (0, 1)$ and $(1, 0) \oplus (0, 1) = (1, 1)$, we have $s = (0, 1)$.

From the problem definition, we immediately see that any deterministic algorithm fails certainly if it has less than $2^{n/2}$ queries, as in this case we can not distinguish between the injective or two-to-one case.

Quantum algorithm. We now present a quantum algorithm that can solve Simon's problem efficiently.

Algorithm 2 (Quantum algorithm for Simon's problem)

Input: two n -qubit registers $|0^n\rangle|0^n\rangle$

- 1: Apply $\boxed{\text{H}}$ to the first n -qubit register to obtain $\frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} |i\rangle|0^n\rangle$.
- 2: Apply the query $\boxed{\text{Q}_{\mathcal{F}}}$ with $|b\rangle = |0^n\rangle$ to obtain $\frac{1}{\sqrt{N}} \sum_{i=0}^{2^n-1} |i\rangle\mathcal{F}(|i\rangle)$.
- 3: Measure the second n -qubit register in the computational basis to obtain some output

$$|a\rangle \otimes |j\rangle = \frac{P_j \left(\frac{1}{\sqrt{N}} \sum |i\rangle\mathcal{F}(|i\rangle) \right)}{\|P_j(\dots)\|}.$$

Note that $|i\rangle\mathcal{F}(|i\rangle) \perp \text{ran } P_j$ for $\mathcal{F}(|i\rangle) \neq |j\rangle$. By assumption, there exist exactly two inputs $|i_0\rangle$ and $|i_1\rangle = |i_0 \oplus s\rangle$ with $\mathcal{F}(|i_k\rangle) = |j\rangle$, $k = 0, 1$. Hence,

$$|a\rangle = \frac{1}{\sqrt{2}}(|i_0\rangle + |i_0 \oplus s\rangle).$$

- 4: Apply $\boxed{\text{H}}$ to the first n -qubit register to obtain

$$\frac{1}{\sqrt{2N}} \sum_{j=0}^{2^n-1} [(-1)^{i_0 \cdot j} + (-1)^{(i_0 \oplus s) \cdot j}] |j\rangle.$$

- 5: Measure in computational basis:
 $|j\rangle$ has non-zero amplitude if $(i_0 \oplus s) \cdot j = i_0 \cdot j \iff s \cdot j = 0 \pmod{2}$.
Thus, we obtain a random element $j^{(1)}$ of the set $\{j : s \cdot j = 0 \pmod{2}\}$.
- 6: Repeat the procedure to obtain $n - 1$ linear independent elements $j^{(1)}, \dots, j^{(n-1)}$ with $j^{(i)} \cdot s = 0 \pmod{2}$ or in other words the linear system

$$\begin{pmatrix} j^{(1)} \\ \vdots \\ j^{(n-1)} \end{pmatrix} s = 0 \pmod{2}$$

- 7: Solve linear system mod 2 on classical computer in $\mathcal{O}(n^3)$.
-

Remark 29. Note that $\#\text{span}\{j^{(1)}, \dots, j^{(k)}\} \leq 2^k$. Hence, with probability $\frac{2^n - 2^k}{2^n} = 1 - 2^{k-n} \geq \frac{1}{2}$ for $k \leq n - 1$, we find a linear independent vector $j^{(k+1)}$.

Complexity. Algorithm 2 requires $\mathcal{O}(n)$ qubits, $\mathcal{O}(n)$ gates, and $\mathcal{O}(n)$ iterations with high probability and additional $\mathcal{O}(n^3)$ effort on a classical computer.

Randomized algorithms for Simon's problem. As previously discussed, a deterministic classical algorithm at least needs $2^{n/2}$ queries. We now show that randomized algorithms on classical computers do not fare much better for this problem.

Lemma 30. Any (randomized) classical algorithm with less than $\delta 2^{n/2}$ queries to \mathcal{F} fails with probability $\geq \exp(-(5/4)\delta^2)$.

Proof. Every algorithm generates a sequence of queries x_1, \dots, x_n with $\underbrace{\mathcal{F}(x_1), \dots, \mathcal{F}(x_n)}_{=: y_1}, \dots, \underbrace{\mathcal{F}(x_n)}_{=: y_n}$.

If all y_i are distinct, the algorithm can't distinguish between $s = 0$ and $s \neq 0$.

Assume all y_1, \dots, y_k with $k < n - 1$ are distinct. Then the algorithm chooses x_{k+1} based on some probability measure μ on $\{0, 1\}^n$ [in the det. case, μ is a delta distribution].

$$\begin{aligned} \sum_{k=1}^{\ell} \sum_{s \in \{0,1\}^n} \sum_{i=1}^k \mu(x_i \oplus s) &= \sum_{k=1}^{\ell} \sum_{i=1}^k \underbrace{\sum_{s \in \{0,1\}^n} \mu(x_i \oplus s)}_{=1} = \frac{\ell(\ell+1)}{2} \\ \implies \exists s \in \{0, 1\}^n : \sum_{k=1}^{\ell} \sum_{i=1}^k \mu(x_i \oplus s) &\leq \frac{\ell(\ell+1)}{2^{n+1}} \leq \frac{1}{2} \quad \text{for } \ell(\ell+1) \leq 2^n. \end{aligned}$$

$p_i \leq 1/2$

Hence, $\mathbb{P}(y_1, \dots, y_{k+1} \text{ distinct}) = \underbrace{\mathbb{P}(y_{k+1} \notin \{y_1, \dots, y_k\} | y_1, \dots, y_k \text{ distinct})}_{\geq 1-p_i} \cdot \mathbb{P}(y_1, \dots, y_k \text{ distinct})$.

Iterate the argument to obtain

$$\mathbb{P}(y_1, \dots, y_{\ell} \text{ distinct}) \geq \prod_{i=1}^{\ell} (1 - p_i).$$

Taking logarithms shows

$$\begin{aligned} \log \left(\prod_{i=1}^{\ell} (1 - p_i) \right) &= \sum_{i=1}^{\ell} \log(1 - p_i) \stackrel{(p_i \leq \frac{1}{2})}{\geq} -\frac{5}{4} \sum_{i=1}^{\ell} p_i \\ &\geq -\frac{5}{4} \frac{\ell(\ell+1)}{2^{n+1}} \\ \implies \mathbb{P}(y_1, \dots, y_{\ell} \text{ distinct}) &\geq e^{-\frac{5}{4} \frac{\ell(\ell+1)}{2^{n+1}}}. \end{aligned}$$

Choosing $\ell + 1 = \delta 2^{n/2}$, which satisfies $\ell(\ell + 1) \leq 2^n$, shows the claimed lower bound $\mathbb{P}(y_1, \dots, y_{\ell} \text{ distinct}) \geq \exp(-(5/4)\delta^2)$. \square

Remark 31. For some $1 - p \leq \xi \leq p$,

$$\log(1 - p) = 0 + \frac{1}{1}(-p) - \frac{1}{2\xi^2}(-p)^2 \geq -p - \frac{p^2}{2} = -p \left(1 + \frac{p}{2}\right) \geq -\frac{5}{4}p.$$

4.3. The quantum Fourier Transform. In the following, we introduce a quantum version of the discrete Fourier transform, which is oftentimes a very useful subroutine in other quantum algorithms.

Discrete Fourier Transform (DFT). We recall the classical DFT: For x_0, \dots, x_{n-1} , define

$$\hat{x}_k := \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \exp\left(-\frac{2\pi i}{N}kj\right)$$

with inverse transformation

$$x_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \hat{x}_j \exp\left(\frac{2\pi i}{N}kj\right).$$

In matrix notation we get

$$\widehat{X} = F_N X \quad \text{with } F_N \in \mathbb{R}^{N \times N}$$

$$(F_N)_{kj} := \frac{1}{\sqrt{N}} \exp\left(-\frac{2\pi i}{N}kj\right)$$

and F_N is a unitary matrix.

Fast Fourier Transform (FFT). Assuming $N = 2^n$, we see that

$$\widehat{x}_k = \frac{1}{\sqrt{2}} \left(\frac{1}{\sqrt{N/2}} \sum_{j=0, j \text{ even}}^{N-1} x_j \exp\left(-\frac{2\pi i}{N/2}k\frac{j}{2}\right) \right. \\ \left. + \exp\left(-\frac{2\pi i}{N}k\right) \frac{1}{\sqrt{N/2}} \sum_{j \text{ odd}} \exp\left(-\frac{2\pi i}{N/2}k\frac{j-1}{2}\right) \right).$$

This allows to split the DFT into 2 DFTs of half size. Proceeding further with this splitting for each auf the 2 sums leads to an algorithm with $\mathcal{O}(N \log N)$ complexity.

Quantum Fourier Transform (QFT). The QFT maps a state $|x\rangle = \sum_{j=0}^{N-1} \widehat{x}_j |j\rangle$ to $|y\rangle = \sum_{j=0}^{N-1} x_j |j\rangle$ where x_j is given by the classical inverse DFT (we implicitly assume that the DFT can be applied to everything here).

If $|x\rangle = |j_0\rangle$ then $x_j = \delta_{jj_0}$ and hence

$$\widehat{x}_k = \frac{1}{\sqrt{N}} \exp\left(\frac{2\pi i}{N}kj_0\right).$$

This leads to the QFT

$$|j\rangle \xrightarrow{QFT} \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \exp\left(\frac{2\pi i}{N}kj\right) |k\rangle = F_N |j\rangle.$$

To implement F_N efficiently, we use the binary representation of k to write

$$\frac{k}{N} = \frac{k}{2^n} = \sum_{\ell=1}^n k_\ell 2^{-\ell}.$$

Therefore,

$$F_N |j\rangle = \frac{1}{\sqrt{2^n}} \sum_{k \in \{0,1\}^n} \underbrace{\exp\left(2\pi i j \sum_{\ell=1}^n k_\ell 2^{-\ell}\right)}_{\otimes_{\ell=1}^n \exp(-2\pi i j k_\ell 2^{-\ell}) |k_\ell\rangle} |k_1 \dots k_n\rangle \\ = \bigotimes_{\ell=1}^n \left(|0\rangle + \exp\left(\frac{2\pi i j}{2^\ell}\right) |1\rangle \right) \frac{1}{\sqrt{2}}.$$

Furthermore, note that because of

$$\exp\left(2\pi i j / 2^\ell\right) = \underbrace{\exp\left(2\pi i \sum_{m=1}^{n-\ell} j_m 2^{n-m-\ell}\right)}_{=1} \exp\left(2\pi i \sum_{m=n-\ell+1}^n j_m 2^{n-m-\ell}\right),$$

the first $n - \ell$ significant bits of j do not matter.

To implement

$$\frac{1}{\sqrt{2}} \left(|0\rangle + \exp \left(2\pi i \sum_{m=n-\ell+1}^n j_m 2^{n-m-\ell} \right) |1\rangle \right)$$

we use the $\boxed{R_s}$ gate (also called \boxed{P} in IBM q-composer) given by the matrix

$$\begin{pmatrix} 1 & 0 \\ 0 & \exp \left(\frac{2\pi i}{2^s} \right) \end{pmatrix}$$

and we need to apply that consecutively as shown in fig. 12 starting from the $n - \ell + 1$ -bit.

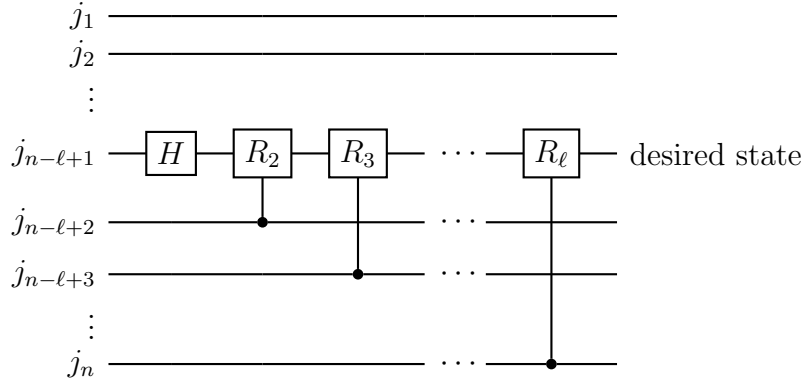


FIGURE 12. Quantum circuit for QFT.

Complexity of QFT. The QFT needs n qubits and $\mathcal{O}(n)$ gates per qubit, which adds up to $\mathcal{O}(n^2)$ gates. This is an exponential speed-up over the FFT with $\mathcal{O}(n2^n)$ operations.

Remark 32. *Strictly speaking, QFT does something different than DFT. The state*

$$\text{QFT}(|x\rangle) = \sum_{j=0}^{N-1} x_j |j\rangle$$

can only be accessed via measurement and hence will collapse to some $|j'\rangle$ with a certain probability. We will see that this is still very useful.

Remark 33. $\boxed{R_s}$ gates don't do very much for large s . One can show that $\mathcal{O}(n \log n)$ gates suffice if one accepts a small error probability.

Remark 34. *Reversing the order of the gates and using the adjoint gates gives an efficient implementation of the inverse QFT F_N^{-1} .*

4.4. Application of QFT: Phase Estimation. Let $(V, (\cdot, \cdot)_V)$ be a Hilbert space with $\dim V = N = 2^n$. Suppose that we have a unitary operator $U : V \rightarrow V$ with eigenvector ψ , i.e.,

$$U\psi = e^{2\pi i\Phi}\psi \quad \text{for some } \Phi \in [0, 1).$$

Assume that $\Phi = \sum_{j=1}^n \Phi_j 2^{-j}$ can be written with n bits.

Since U is an operator on a 2^n -dimensional Hilbert space, classical computation of $(U\Psi, \Psi)_V$ requires at least $\mathcal{O}(2^n)$ operations.

Quantum algorithm. We now consider a quantum algorithm for efficient evaluation of $(U\Psi, \Psi)_V$, which is called *quantum phase estimation* in literature.

Algorithm 3 (Quantum Phase estimation)

Input: $|0^n\rangle|\Psi\rangle$

- 1: Apply $H^{\otimes n}$ to the first n qubits to obtain $\frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} |j\rangle|\Psi\rangle$.
- 2: Apply $|j\rangle|\Psi\rangle \mapsto |j\rangle U^j |\Psi\rangle$ to obtain

$$\frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2\pi i j \Phi} |j\rangle |\Psi\rangle.$$

Note that the first n -qubit register satisfies

$$\frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2\pi i j \Phi} |j\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2\pi i j N \Phi / N} |j\rangle = F_N(|N\Phi\rangle).$$

- 3: Apply IQFT to the first n -qubit register to obtain

$$F_N^{-1} \left(\frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2\pi i j \Phi} |j\rangle \right) = |N\Phi\rangle$$

- 4: Measure in computational basis to obtain $N\Phi$ and hence Φ .
-

The last formula in step 2 might be a bit confusing since suddenly $\Phi \in \mathbb{R}$ is interpreted as an element of the vector space in which Ψ is contained. However, $N\Phi = \sum_{j=1}^n \Phi_j 2^{n-j}$ is a basis element and hence makes sense.

In terms of implementation, the question is how to realize step 2 of the quantum phase estimation algorithm. This can be done as shown in the circuit diagram in fig. 13 below.

Nonetheless, we will need to assume that $[U^{2^k}]$ can be implemented efficiently. This might not be true in general, but is in the applications below.

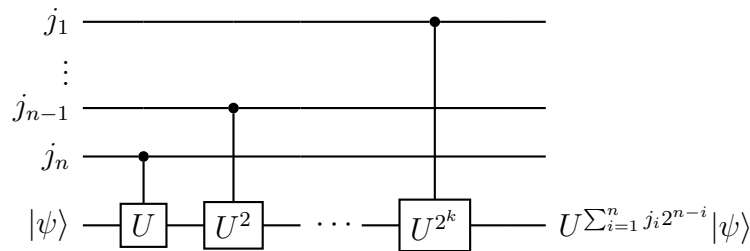


FIGURE 13. Quantum circuit for quantum phase estimation.

Remark 35. Note that the input doesn't need to be a single eigenvector. Let $|\Psi\rangle, |\Psi'\rangle$ be two eigenvectors with corresponding phases Φ, Φ' . Linearity implies that the input $1/\sqrt{2}(|\Psi\rangle + |\Psi'\rangle)$ will produce the output $1/\sqrt{2}(|N\Phi\rangle + |N\Phi'\rangle)$. A measurement will produce either Φ or Φ' with equal probability.

If Φ requires more than n bits, the final state is a small perturbation δ of $F_N(|N\Phi_n\rangle)$ with

$$|\delta| = \left| \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \underbrace{[e^{2\pi i j \Phi} - e^{2\pi i j \Phi_n}]}_{\mathcal{O}(2^{-n})} |j\rangle \right| / \left| \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2\pi i j \Phi} |j\rangle \right| \simeq \mathcal{O}(2^{-n}).$$

4.5. Shor's integer factorization. Now, we have all the tools to formulate and analyze the arguably historically first quantum algorithm that can be applied for a mathematical problem with many real world applications, the question on fast integer factorization.

Problem formulation. Given $N \in \mathbb{N} \setminus \{\text{primes}\}$, find an integer $k \in \mathbb{N}$, $1 < k < N$ such that $N/k \in \mathbb{N}$.

For this problem, we note that:

- The security of most current cryptography-systems is based on the fact that integer factorization is hard.
- A number $N \in \mathbb{N}$ can be defined using $\log_2(N)$ bits. Thus, polynomial complexity of algorithms in this context means $\mathcal{O}((\log_2 N)^p)$ operations.
- There exist efficient classical algorithms to check whether N is a prime or power of a prime (see, e.g. AKS-test or BPSW-test).
- The best known classical algorithm for integer factorization is the general number field sieve with runtime $\mathcal{O}(\exp((\log_2 N)^{1/3}(\log_2 \log_2 N)^{2/3}))$. Note that the runtime is a conjecture!
- It is not known that classical algorithms can not be faster. Latest paper which (falsely) claimed an $\mathcal{O}((\log N)^p)$ -algorithm was by Schnour (Bonn) in 2021.

Our goal is to formulate and analyze a quantum algorithm, developed by the mathematician Peter Shor in 1994, that achieves integer factorization in polynomial complexity.

Reduction to period finding. The key for the algorithm is a reformulation of the factorization problem as follows:

- (1) Choose a random $x \in \{2, \dots, N-1\}$ such that for the greatest common divisor there holds $\gcd(x, N) = 1$. This condition can be checked efficiently with the Euclidean algorithm.
In other words $x \in (\mathbb{Z}/N\mathbb{Z})^\times$, which denotes the multiplicative group mod N .
- (2) Using Lemma 36 below, we can infer that x has period r with $x^r \bmod N = 1$.
- (3) With probability at least $1/2$, there holds that r is even and $x^{r/2} \pm 1 \not\equiv 0 \pmod N$.
- (4) This implies that $(x^{r/2} - 1)(x^{r/2} + 1) \equiv x^r - 1 \equiv 0 \pmod N$.

We have thus found $\gcd(x^{r/2} - 1, N)$ and $\gcd(x^{r/2} + 1, N)$ which are (due to step (3)) non-trivial factors of N .

In the following, we will prove these statements and provide an algorithm on how to calculate the period r .

We start with the proof of statement (2).

Lemma 36. *Every $x \in (\mathbb{Z}/N\mathbb{Z})^\times$ has a period r , which is the minimal $r \in \mathbb{N}$ such that $x^r \bmod N = 1$.*

Proof. Consider the set $S = \{1, x^1, \dots\} \subset (\mathbb{Z}/N\mathbb{Z})^\times$. Since $(\mathbb{Z}/N\mathbb{Z})^\times$ is finite, there exists $j \neq k \in \mathbb{N}$ with $x^j = x^k \pmod{N}$. W.l.o.g. assume $j < k$. Then, $x^{k-j} \equiv 1 \pmod{N}$. \square

We recall that the order of an element x in a multiplicative group is the smallest integer k such that $x^k = 1$. For a multiplicative group $(\mathbb{Z}/n\mathbb{Z})^\times$ this is given by the Euler totient function ϕ defined by $\phi(n) := \#\{1 \leq k \leq n : \gcd(k, n) = 1\}$. There holds

$$\phi(n) = n \prod_{\substack{p|n \\ p \text{ prime}}} \left(1 - \frac{1}{p}\right) \implies \phi(p^\alpha) = p^\alpha \left(1 - \frac{1}{p}\right) = p^{\alpha-1}(p-1) \quad \text{for } \alpha \in \mathbb{N}.$$

Lemma 37 (Chinese remainder Theorem). *Let $m_1, \dots, m_n \in \mathbb{N}$ with $\gcd(m_i, m_j) = 1$ for $i \neq j$. Then, the problem:*

$$\begin{aligned} & x = a_1 \pmod{m_1} \\ \text{Find } x \text{ s.t.: } & \quad \vdots \\ & x = a_n \pmod{m_n} \end{aligned}$$

is solvable and any two solutions are equal mod $M = m_1 m_2 \dots m_n$.

Proof. Define $M_i := M/m_i$. Then, $\gcd(M_i, m_i) = 1$ and M_i has a multiplicative inverse mod m_i denoted by N_i .

Now, $x = \sum_i a_i M_i N_i$ is a solution to the given problem, since $M_i N_i \equiv 1 \pmod{m_i}$ and $M_i N_i \equiv 0 \pmod{m_j}$ for $i \neq j$.

Two solutions x, x' satisfy $x - x' \equiv 0 \pmod{m_i}$ for all i and hence $M = m_1 m_2 \dots m_n$ divides $x - x'$ since the m_i are coprime. \square

Now, we show the statement (3) above, for which we need a lemma first.

Lemma 38. *Let $p > 2$ be a prime, $\alpha \in \mathbb{N}$, and let 2^d be the maximal power of 2 dividing $\phi(p^\alpha)$. Let x be a randomly chosen element in $(\mathbb{Z}/p^\alpha\mathbb{Z})^\times$. Then,*

$$\mathbb{P}(2^d \text{ divides order of } x) = \frac{1}{2}.$$

Proof. If $\phi(p^\alpha) = p^{\alpha-1}(p-1)$ is even then $2^d \geq 1$.

Let g denote the generator of $(\mathbb{Z}/p^\alpha\mathbb{Z})^\times$. Then $x \equiv g^k \pmod{p^\alpha}$ for some $k \in \{1, \dots, \phi(p^\alpha)\}$. Let r denote the order of x . Then, we have for

- k odd: $g^{kr} \equiv 1 \pmod{p^\alpha} \implies \phi(p^\alpha) | kr$ since $\phi(p^\alpha)$ is minimal with $g^{\phi(p^\alpha)} = 1$. Furthermore, since k is odd, $2^d | r$;
- k even: $g^{k\phi(p^\alpha)/2} \equiv 1 \pmod{p^\alpha} \implies r | \frac{\phi(p^\alpha)}{2}$ since r is minimal with $x^r \equiv g^{kr} \equiv 1 \pmod{p^\alpha}$. Hence, $2^d \nmid r$.

Consequently, for exactly half of $x \in (\mathbb{Z}/p^\alpha\mathbb{Z})^\times$, we have $x = g^k$ with k even/odd. \square

Lemma 39. *Let $N = p_1^{\alpha_1} \dots p_m^{\alpha_m}$ be a prime factorization of $N \notin 2\mathbb{N}$ (odd). Let x be randomly chosen in $(\mathbb{Z}/N\mathbb{Z})^\times$ with order r . Then,*

$$\mathbb{P}(r \text{ is even and } x^{\frac{r}{2}} \not\equiv -1 \pmod{N}) \geq 1 - 2^{-m+1}.$$

Proof. Following Lemma 37, choosing x randomly is equivalent to choosing x_j randomly in $(\mathbb{Z}/p_j^{\alpha_j}\mathbb{Z})^\times$ with $x = x_j \pmod{p_j^{\alpha_j}}$ for $j = 1, \dots, m$ since $x \mapsto (x_1, \dots, x_m)$ is one-to-one. Let r_j be the order of $x_j \pmod{p_j^{\alpha_j}}$ and let 2^{d_j} be the maximal power of 2 dividing r_j . We will show that

$$r \text{ is odd or } x^{\frac{r}{2}} \equiv -1 \pmod{N} \implies d_1 = d_2 = \dots = d_m.$$

If this is the case, the following argument concludes the proof: Let d'_j be maximal such that $2^{d'_j}$ divides $\phi(p_j^{\alpha_j})$. Lemma 38 states that

$$\mathbb{P}(d_j = d'_j) = \frac{1}{2} \implies \mathbb{P}(d_j = k) \leq \frac{1}{2} \quad \forall k \in \mathbb{N}.$$

Furthermore, since d_j are independent, we have

$$\mathbb{P}(d_1 = \dots = d_m) \leq 2^{-m+1}.$$

It remains to show that $d_1 = \dots = d_m$.

Case 1: r is odd and $x^r \equiv 1 \pmod{N}$. Then,

$$x^r \equiv 1 \pmod{p_j^{\alpha_j}} \implies r_j | r \implies r_j \text{ is odd} \implies d_j = 0.$$

Case 2: r is even and $x^{r/2} = -1 \pmod{N}$. Then,

$$x^{r/2} = -1 \pmod{p_j^{\alpha_j}}.$$

If r_j would divide $r/2$, we would have

$$\left. \begin{aligned} x^{r/2} = x^{kr_j} &= -1 \pmod{p_j^{\alpha_j}} \\ x^{kr_j} &= x_j^{kr_j} \pmod{p_j^{\alpha_j}} \\ x^{kr_j} &= 1 \pmod{p_j^{\alpha_j}} \end{aligned} \right\} \implies r_j \nmid \frac{r}{2}.$$

On the other hand r_j divides r . Hence, largest power of 2 dividing r must be equal to d_j . □

Note that, if r is the period of $x \pmod{N}$ and r is even, then $x^{r/2} \neq 1 \pmod{N}$ with

$$\mathbb{P}(r \text{ even}, x^{r/2} \neq \pm 1 \pmod{N}) \geq 1 - 2^{-m+1} \geq \frac{1}{2},$$

if $m \geq 2$, i.e., if N is not a prime or power of prime.

Now, define $U|y\rangle := |xy \pmod{N}\rangle$. Note that $|j\rangle \mapsto |xj \pmod{N}\rangle$ is bijective since $xj = xj' \pmod{N}$ and $x(j-j') = 0 \pmod{N}$ and thus $\gcd(x, N) \neq 1$. Therefore U is a permutation of basis elements and hence unitary.

Lemma 40. *Let r denote the period of $x \in (\mathbb{Z}/N\mathbb{Z})^\times$. Then,*

$$|u_s\rangle := \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \exp\left(\frac{-2\pi i s k}{r}\right) |x^k \pmod{N}\rangle$$

are eigenvectors of U with eigenvalues

$$\lambda_s := \exp\left(\frac{2\pi i s}{r}\right), \quad s = 0, \dots, r-1.$$

Proof. Note that $x^r = x^0 \pmod{N}$ and $\exp(-2\pi i s) = \exp(0)$. Hence,

$$\begin{aligned} U|u_s\rangle &= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \exp\left(\frac{-2\pi i s k}{r}\right) |x^{k+1} \pmod{N}\rangle \\ &= \lambda_s \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \exp\left(\frac{-2\pi i s (k+1)}{r}\right) |x^{k+1} \pmod{N}\rangle \\ &= \lambda_s |u_s\rangle. \end{aligned}$$

□

Note that there holds

$$\begin{aligned}
\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle &= \frac{1}{r} \sum_{s=0}^{r-1} \sum_{k=0}^{r-1} \exp\left(\frac{-2\pi i s k}{r}\right) |x^k \bmod N\rangle \\
&= \frac{1}{r} \sum_{k=0}^{r-1} \underbrace{\sum_{s=0}^{r-1} \exp\left(\frac{-2\pi i s k}{r}\right)}_{\begin{cases} 0 & k \neq 0 \\ r & k = 0 \end{cases}} |x^k \bmod N\rangle \\
&= |x^0 \bmod N\rangle = |1\rangle.
\end{aligned}$$

Consequently, quantum phase estimation for U with input $|1\rangle$ produces the state

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |2^n \frac{s}{r}\rangle.$$

Due to $r \leq N$, s/r can be exactly represented with n bits.

To efficiently implement

$$U^{2^k} |y\rangle = |xy^{2^k} \bmod N\rangle$$

we use the fact that

$$y^{2^k} = \left(\dots \left((y^2)^2\right) \dots\right)^2.$$

Hence, $|xy^{2^k} \bmod N\rangle$ requires 1 multiplication and k squares mod N . Thus, it can be implemented as in classical circuits with XOR & AND gates.

Quantum algorithm. Note that the period finding is the only quantum part of Shor's algorithm. Let $N \leq 2^n$.

Algorithm 4 (Shor's period finding algorithm)

- 1: Prepare $|\Psi\rangle = |1\rangle = \underbrace{|0 \dots 01\rangle}_n$ and U as before
- 2: Use quantum phase estimation with U and $|\Psi\rangle$ to obtain the state

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |2^n \frac{s}{r}\rangle$$

- 3: Measurement gives a random number s/r from the set

$$\left\{ \frac{1}{r}, \dots, \frac{r-1}{r} \right\} \subseteq \left\{ \frac{i}{2^n} : i = 0, \dots, 2^n - 1 \right\}$$

- 4: Write $s/r = s_0/r_0$ with $\gcd(s_0, r_0) = 1$. If s and r were coprime already, we have $r = r_0$ and found the period. Otherwise, repeat.
-

Remark 41. There are $\mathcal{O}\left(\frac{r}{\log \log r}\right)$ Euler totient function numbers $1 \leq s < r$ with $\gcd(s, r) = 1$. Therefore, the success-probability of algorithm 4 is $\mathcal{O}\left(\frac{1}{\log \log r}\right)$. Furthermore, since

$$\frac{1}{\log \log r} \geq \frac{1}{\log \log N} \geq \frac{1}{\log n},$$

Algorithm 4 requires on average $\log n$ repetitions.

Remark 42. Implicitly, we used QFT to find the frequency of $x^k \bmod N$, i.e., the period x .

4.6. Grover's search algorithm. The following quantum algorithm for search in an unstructured list is considered to be the historically second major quantum algorithm and is attributed to the computer scientist Lov Grover in 1996.

Problem formulation Given a function $\mathcal{F} : \{0, 1\}^n \rightarrow \{0, 1\}$, the goal is to find $x \in \{0, 1\}^n$ with $\mathcal{F}(x) = 1$ or determine that $\mathcal{F} = 0$.

Identifying $x \in \{0, 1\}^n$ with $i = 0, \dots, 2^n - 1$, we recall the query

$$Q_{\mathcal{F}}(|i-\rangle) = (-1)^{\mathcal{F}(i)} |i-\rangle.$$

Define the *Grover diffusion operator*

$$U_s := 2 \underbrace{|s\rangle\langle s|}_{\substack{\text{projection} \\ \text{onto } |s\rangle}} - \text{id},$$

where $|s\rangle$ denotes the uniform state

$$|s\rangle := \frac{1}{N} \sum_{j=0}^{N-1} |j\rangle.$$

Note that $|s\rangle = H^{\otimes n} |0^n\rangle$ and hence

$$U_s = H^{\otimes n} (2|0^n\rangle\langle 0^n| - \text{id}) H^{\otimes n},$$

where $(2|0^n\rangle\langle 0^n| - \text{id})$ corresponds to the matrix

$$\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & -1 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & -1 \end{pmatrix} \in \mathbb{R}^{2^n \times 2^n}.$$

Implementation of the Grover diffusion operator. The following circuit diagram implements

$$-(2|0^n\rangle\langle 0^n| - \text{id}),$$

which is up to the sign what we want.

We note that any global phase change, i.e., a multiplication with $\lambda \in \mathbb{C}$, $|\lambda| = 1$ (and thus the different sign above) is not noticable, as measurements are always based on squared absolute values of the amplitudes.

Algorithm 5 (Grover's algorithm)

1: Apply $H^{\otimes n}$ to obtain

$$|s\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} |j\rangle.$$

2: For $k = 1, \dots, r(N)$ do

a: Apply $Q_{\mathcal{F}}$ to current state $\sum_{j=0}^{N-1} \alpha_j |j\rangle$ to obtain

$$\sum_{j=0}^{N-1} (-1)^{\mathcal{F}(j)} \alpha_j |j\rangle.$$

b: Apply U_s to first n qubits.

3: Measure in computational basis.

Theorem 43. *Let*

$$r(N) = \frac{\arccos\left(\sqrt{\frac{t}{N}}\right)}{2 \arcsin\left(\sqrt{\frac{t}{N}}\right)},$$

then Grover's algorithm finds state $|x\rangle$ with $\mathcal{F}(x) = 1$ with probability at least $1 - \frac{t}{N}$.

Proof. Define

$$|B\rangle := \frac{1}{\sqrt{N-t}} \sum_{\mathcal{F}(j)=0} |j\rangle,$$

where $t = \#\mathcal{F}^{-1}(\{1\})$. Observe that the first n qubits of $|x\rangle \mapsto Q_{\mathcal{F}}(|x\rangle)$ satisfy

$$|x\rangle \mapsto (2|B\rangle\langle B| - \text{id})|x\rangle = U_B(|x\rangle),$$

which can be checked for basis elements $|x\rangle = |j\rangle$.

Define $|\zeta\rangle := \frac{1}{\sqrt{t}} \sum_{\mathcal{F}(j)=1} |j\rangle$ and note that $|s\rangle \in \text{span}\{|\zeta\rangle, |B\rangle\}$, and $U_s, U_B : \text{span}\{|\zeta\rangle, |B\rangle\} \rightarrow \text{span}\{|\zeta\rangle, |B\rangle\}$. Hence, the Grover iteration never leaves the plane $\text{span}\{|\zeta\rangle, |B\rangle\}$ as depicted in Figure 16.

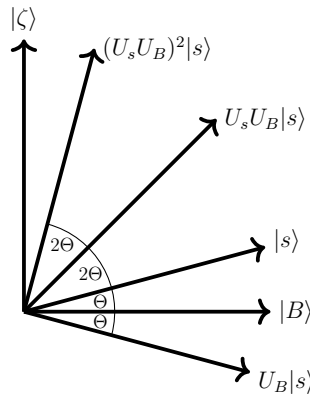


FIGURE 16. Grover iterates in the $|\zeta\rangle, |B\rangle$ -plane.

Each application of $U_s U_B$ rotates a state $|x\rangle \in \text{span}\{|\zeta\rangle, |B\rangle\}$ towards $|\zeta\rangle$ by an angle Θ given by

$$\cos \Theta = \frac{\langle s|B\rangle}{\|s\|\|B\|} = \frac{1}{\sqrt{N}} \frac{1}{\sqrt{N-t}} \sum_{\mathcal{F}(|j\rangle)=0} \langle j|j\rangle = \frac{N-t}{\sqrt{N(N-t)}} = \sqrt{1 - \frac{t}{N}}.$$

Furthermore, this implies

$$\sin^2 \theta = 1 - \cos^2 \theta = 1 - 1 + \frac{t}{N} \implies \sin \theta = \sqrt{\frac{t}{N}}.$$

For large N , we therefore have $\theta \approx \sqrt{t/N}$. Since the angle between $|\zeta\rangle$ and $|s\rangle$ is

$$\cos \alpha = \frac{1}{\sqrt{N}} \frac{1}{\sqrt{t}} \sum_{\mathcal{F}(|j\rangle)=1} \langle j|j\rangle = \sqrt{\frac{t}{N}} \implies r(N) = \text{round} \left(\frac{\arccos \left(\sqrt{\frac{t}{N}} \right)}{2 \arcsin \left(\sqrt{\frac{t}{N}} \right)} \right)$$

to obtain a state $|x\rangle$ with $\langle \zeta|x\rangle \geq \cos \theta$ and $|\langle \zeta|x\rangle|^2 \geq \cos^2 \theta = 1 - (t/N)$. \square

Remark 44. *In practice, we do not know when $|x\rangle$ gets close to $|\zeta\rangle$. But we can measure and check whether $F(x) = 1$. If not, we restart the algorithm. If $t \ll N$ there follows*

$$r(N) = \frac{\arccos \left(\sqrt{\frac{t}{N}} \right)}{2 \arcsin \left(\sqrt{\frac{t}{N}} \right)} \approx \frac{\pi}{4} \sqrt{\frac{N}{t}}.$$

Remark 45. *If $t = 1$, a classical algorithm requires at least N evaluations of \mathcal{F} . Grover's algorithm only needs $\mathcal{O}(\sqrt{N})$ iterations with $\mathcal{O}(n)$ gates.*

Optimality of Grover's algorithm.

Lemma 46. *Any quantum algorithm based on the query $Q_{\mathcal{F}}$ requires at least $\mathcal{O}(\delta\sqrt{N})$ applications of $Q_{\mathcal{F}}$ to succeed with probability of at least δ^2 .*

Proof. Any quantum algorithm starts with some state $|\psi\rangle$ and applies unitary transformations as well as $Q_{\mathcal{F}}$. Consequently, we may write the state after k applications of $Q_{\mathcal{F}}$ as

$$|\psi_k^{\mathcal{F}}\rangle = U_k Q_{\mathcal{F}} U_{k-1} Q_{\mathcal{F}} \cdots U_1 Q_{\mathcal{F}} |\psi\rangle,$$

where U_k, \dots, U_1 are unitary and $Q_{\mathcal{F}}$ is the query. Additionally, we will consider

$$|\psi_k\rangle := U_k U_{k-1} \cdots U_1 |\psi\rangle.$$

Let $\mathcal{F}_j : \{0, 1\}^n \rightarrow \{0, 1\}$ with $\mathcal{F}_j(|j\rangle) = \delta_{ij}$ and define

$$D_k = \sum_{j=0}^{N-1} \|\psi_k^{\mathcal{F}_j} - \psi_k\|^2.$$

Idea: If D_k is small, the evaluation of \mathcal{F} does not make a big difference and it will be hard to find $\mathcal{F}(|i\rangle) = 1$.

Step 1: Show that $D_k \leq 4k^2$ by induction:

For $k = 0$ there holds $D_0 = 0$.

For the induction step $k \mapsto k + 1$, we observe

$$\begin{aligned}
D_{k+1} &= \sum_{j=0}^{N-1} \|U_{k+1} Q_{\mathcal{F}_j} \psi_k^{\mathcal{F}_j} - U_{k+1} \psi_k\|^2 \\
&= \sum_{j=0}^{N-1} \|Q_{\mathcal{F}_j} \psi_k^{\mathcal{F}_j} - \psi_k\|^2 \\
&= \sum_{j=0}^{N-1} \|Q_{\mathcal{F}_j} (\psi_k^{\mathcal{F}_j} - \psi_k) + (Q_{\mathcal{F}_j} - \text{id}) \psi_k\|^2.
\end{aligned}$$

Note that $Q_{\mathcal{F}_j} = \text{id} - 2|j\rangle\langle j| \implies (Q_{\mathcal{F}_j} - \text{id})|\psi_k\rangle = -2|j\rangle\langle j|\psi_k\rangle$ and thus

$$D_{k+1} \leq \sum_{j=0}^{N-1} \|\psi_k^{\mathcal{F}_j} - \psi_k\|^2 + 4\|\psi_k^{\mathcal{F}_j} - \psi_k\| |\langle j|\psi_k\rangle| + 4|\langle j|\psi_k\rangle|^2.$$

Furthermore, Cauchy-Schwarz shows that

$$\begin{aligned}
D_{k+1} &\leq D_k + 4 \left(\sum_{j=0}^{N-1} \|\psi_k^{\mathcal{F}_j} - \psi_k\|^2 \right)^{\frac{1}{2}} \underbrace{\left(\sum_{j=0}^{N-1} |\langle j|\psi_k\rangle|^2 \right)^{\frac{1}{2}}}_{=1} + 4 \underbrace{\langle \psi_k|\psi_k\rangle^2}_{=1} \\
&\leq D_k + 4\sqrt{D_k} + 4.
\end{aligned}$$

Using the induction hypotheses, now leads to $D_k + 4\sqrt{D_k} + 4 \leq 4k^2 + 8k + 4 = 4(k+1)^2$.

This concludes the induction and shows that $D_k \leq 4k^2$.

Step 2: Assume that $|\langle j|\psi_k^{\mathcal{F}_j}\rangle|^2 \geq \delta^2 > 0 \quad \forall j = 1, \dots, N-1$, i.e., the algorithm works with high probability for each input. Replacing $|j\rangle$ with $\exp(i\theta)|j\rangle$ does not change the success probability. Hence, we may assume that $|\langle j|\psi_k^{\mathcal{F}_j}\rangle| = \langle j|\psi_k^{\mathcal{F}_j}\rangle$ and thus

$$\|\psi_k^{\mathcal{F}_j} - j\|^2 = 2 - 2|\langle j|\psi_k^{\mathcal{F}_j}\rangle| \leq 2(1 - \delta).$$

By defining

$$\begin{aligned}
E_k &:= \sum_{j=0}^{N-1} \|\psi_k^{\mathcal{F}_j} - j\|^2 \implies E_k \leq 2N(1 - \delta) \\
F_k &:= \sum_{j=0}^{N-1} \|j - \psi_k\|^2,
\end{aligned}$$

we see that

$$\begin{aligned}
D_k &= \sum_{j=0}^{N-1} \|(\psi_k^{\mathcal{F}^j} - j) + (j - \psi_k)\|^2 \\
&\geq \sum_{j=0}^{N-1} \|\psi_k^{\mathcal{F}^j} - j\|^2 - 2\|\psi_k^{\mathcal{F}^j} - j\|\|j - \psi_k\| - \|j - \psi_k\|^2 \\
&= E_k + F_k - 2 \left(\sum_{j=0}^{N-1} \|\psi_k^{\mathcal{F}^j} - j\|^2 \right)^{\frac{1}{2}} \left(\sum_{j=0}^{N-1} \|j - \psi_k\|^2 \right)^{\frac{1}{2}} \\
&= E_k + F_k - 2\sqrt{E_k F_k} = \left(\sqrt{E_k} - \sqrt{F_k} \right)^2.
\end{aligned}$$

Note that any state $|\Phi\rangle$ satisfies

$$\begin{aligned}
\sum_{j=0}^{N-1} \|\Phi - j\|^2 &= \sum_{j=0}^{N-1} \underbrace{\|\Phi\|^2}_{=1} - 2\langle \Phi | j \rangle + \underbrace{\|j\|^2}_{=1} \\
&\geq 2N - 2 \underbrace{\sqrt{\sum_{j=0}^{N-1} 1} \sqrt{\sum_{j=0}^{N-1} |\langle \Phi | j \rangle|^2}}_{=1} = 2(N - \sqrt{N}).
\end{aligned}$$

This implies $F_k \geq 2(N - \sqrt{N})$ and hence $F_k \geq E_k$ for sufficiently large N and

$$\begin{aligned}
\sqrt{F_k} - \sqrt{E_k} &\geq \sqrt{2(N - \sqrt{N})} - \sqrt{2N(1 - \delta)} = \frac{2\delta N - 2\sqrt{N}}{\sqrt{2(N - \sqrt{N})} + \sqrt{2N(1 - \delta)}} \\
&\geq \frac{2\delta N - 2\sqrt{N}}{2\sqrt{2N}} = \frac{\delta}{\sqrt{2}}\sqrt{N} - \frac{1}{\sqrt{2}}.
\end{aligned}$$

Therefore, $D_k \geq (\sqrt{F_k} - \sqrt{E_k})^2 \approx \delta^2 N$ and since $D_k \leq 4k^2$, we have $k \approx \delta\sqrt{N}$. \square

Example 47. *Qiskit example* The link to this qiskit example can be found on the web page for this lecture. Let $N = 16$, $n = 4$, and

$$\mathcal{F}(x) = \mathcal{F}(x_0, \dots, x_3) = x_0 \text{ AND } x_1 \text{ AND } x_2 \text{ AND } x_3 = \delta_{x=(1,1,1,1)}$$

which implies $t = 1$.

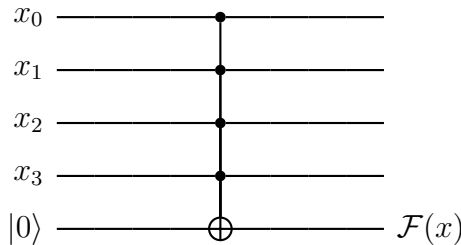


FIGURE 17. \mathcal{F} as quantum circuit.

Since

$$\frac{\arccos\left(\sqrt{\frac{1}{16}}\right)}{2 \arcsin\left(\sqrt{\frac{1}{16}}\right)} \approx 2.6083,$$

the optimal $r(N)$ is 3 and the probability of success is therefore at least $1 - 1/N = 15/16 \approx 0.9375$.

5. NUMERICAL ALGORITHMS ON QUANTUM COMPUTERS

In this section, we revisit standard topics in numerical analysis such as numerical integration, solving linear systems of equations, finding zeros of a non-linear function and numerical solution of ODEs and discuss how they can be formulated on quantum computers. Moreover, we discuss whether quantum advantage can be achieved for these problems.

5.1. Numerical Integration. We start with the problem of calculating an integral by means of quadrature

$$\int_0^1 f \approx \frac{1}{N} \sum_{i=0}^{N-1} \omega_i f(\xi_i).$$

with quadrature nodes $\xi_i \in [0, 1]$ and weights ω_i .

In the following we consider a very easy quadrature rule, so called sampling quadrature, where $\omega_i = \frac{1}{N}$ for all i and ξ_i are uniformly chosen in $[0, 1]$.

Problem formulation. Given $f : \{0, 1\}^n \rightarrow [-1, 1]$, compute

$$\frac{1}{N} \sum_{i=0}^{N-1} f(|i\rangle),$$

where $|i\rangle$ is again identified with its unique bit-string in $\{0, 1\}^n$. As each bit-string also identifies with a number $\xi \in [0, 1]$, we indeed have a quadrature formula on $[0, 1]$.

Quantum algorithm. The following quantum algorithm is called *quantum super sampling* and provides a realization of the sampling quadrature in our problem formulation.

Consider the Oracle Q_f defined by

$$Q_f : |0\rangle \otimes |i\rangle \mapsto \left[\sqrt{1 - f(i)}|0\rangle + \sqrt{f(i)}|1\rangle \right] \otimes |i\rangle.$$

- 1: Start with $|0^P\rangle \otimes |0\rangle \otimes |0^n\rangle$
- 2: Apply $\boxed{\text{H}}^{\otimes P} \otimes \text{id} \otimes \boxed{\text{H}}^{\otimes n}$ to obtain

$$\frac{1}{\sqrt{PN}} \sum_{i=0}^{N-1} \sum_{m=0}^{P-1} |m\rangle \otimes |0\rangle \otimes |i\rangle$$

- 3: Apply $\boxed{Q_f}$ to obtain

$$\frac{1}{\sqrt{PN}} \sum_{i=0}^{N-1} \sum_{m=0}^{P-1} |m\rangle \otimes \left[\sqrt{1 - f(i)}|0\rangle + \sqrt{f(i)}|1\rangle \right]$$

- 4: Define

$$\begin{aligned} |G\rangle &:= \sqrt{\frac{\sum f(i)}{N}} \sum_{i=0}^{N-1} \sqrt{f(i)}|1\rangle \otimes |i\rangle \\ |B\rangle &:= \sqrt{\frac{\sum 1 - f(i)}{N}} \sum_{i=0}^{N-1} \sqrt{1 - f(i)}|0\rangle \otimes |i\rangle \\ \bar{f} &:= \frac{1}{N} \sum f(i) \end{aligned}$$

Note that

$$\begin{aligned} U_s &:= (1|0\rangle\langle 0| - \text{id}) \otimes \text{id} \\ U_s|B\rangle &= |B\rangle \\ U_s|\zeta\rangle &= -|G\rangle. \end{aligned}$$

Furthermore,

$$U_\psi = (2|\psi\rangle\langle\psi| - \text{id})$$

with $|\psi\rangle := \sqrt{1-\bar{f}}|B\rangle + \sqrt{\bar{f}}|\zeta\rangle$. Section 5.1 provide a quantum circuit to implement the map $U : |0\rangle \oplus |0^n\rangle \mapsto |\psi\rangle$, and hence

$$U_\psi = U^{-1}(2|0^{n+1}\rangle\langle 0^{n+1}| - \text{id})U$$

as in Grover's algorithm.

5: For $\bar{f} := 1/N \sum f(i)$, the state reads

$$\frac{1}{\sqrt{P}} \sum_{m=0}^{P-1} |m\rangle \otimes \left[\sqrt{1-\bar{f}}|B\rangle + \sqrt{\bar{f}}|\zeta\rangle \right]$$

6: Apply Grover iteration $U_\psi U_s$ m -times to second $n+1$ qubits to obtain

$$\frac{1}{\sqrt{P}} \sum_{m=0}^{P-1} |m\rangle \otimes (U_\psi U_s)^n \left[\sqrt{1-\bar{f}}|B\rangle + \sqrt{\bar{f}}|\zeta\rangle \right]$$

Explanation: Let $\sin \theta = \sqrt{\bar{f}}$, then $|\psi\rangle = \cos \theta|B\rangle + \sin \theta|\zeta\rangle$ and $(U_\psi U_s)^m |\psi\rangle = \cos(2m+1)\theta|B\rangle + \sin(2m+1)\theta|\zeta\rangle$. U_s acts like a reflection across $|B\rangle$ as shown in fig. 18.

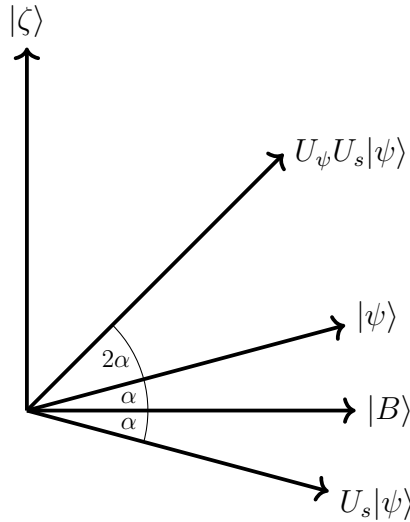


FIGURE 18. $U_\psi U_s$ and U_s acting on $|\psi\rangle$

Note that although the m -times application of $U_\psi U_s$ is as in algorithm 3, the cost is in general $\mathcal{O}(Pm)$.

7: $|\psi\rangle$ rotates with rate 2θ in the $|B\rangle, |\zeta\rangle$ -plane. Use *QFT* to obtain the state

$$\frac{1}{\sqrt{P}} \sum_{m=0}^{P-1} |m\rangle \otimes [\cos(2m+1)\theta|B\rangle + \sin(2m+1)\theta|\zeta\rangle].$$

Measure the last $n+1$ qubits in $\{|\zeta\rangle, |B\rangle\}$ -basis to obtain

$$\frac{1}{C} \sum_{m=0}^{P-1} \underbrace{\sin[(2m+1)\theta]}_{\hat{x}_m} |m\rangle$$

where $C = 1/P \sum_{m=0}^{P-1} \sin[(1m+1)\theta]^2$.

8: Under the assumption that $\theta = \pi\theta_0/P$, $\theta_0 \in k$, apply *QFT* to obtain

$$\frac{1}{C} \sum_{m=0}^{P-1} x_m |m\rangle,$$

where x_m is given by *IFT* of \hat{x}_m , i.e.

$$\begin{aligned} x_m &= \frac{1}{P} \sum_{k=0}^{P-1} \hat{x}_k e^{\frac{2\pi i m k}{P}} = \frac{1}{\sqrt{P}} \frac{P-1}{2} \frac{1}{2i} \begin{pmatrix} i(2k+1)\frac{\theta_0}{P} & -i(2k+1)\frac{\theta_0}{P} \\ e & -e \end{pmatrix} e^{\frac{2\pi i m k}{P}} \\ &= e^{i\frac{\theta_0}{P}} \frac{1}{2i} \begin{cases} \sqrt{P} & m = \theta_0 \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

9: Final measurement produces $|m\rangle = |\theta_0\rangle$ and hence $\bar{f} = \sin(\pi\theta_0/P)^2$.

Remark 48. *The implementation of Q_f is not clear. If $f : \{0, 1\}^n \rightarrow \{0, 1\}$, we can use a controlled not-gate to implement the query as shown in fig. 19.*

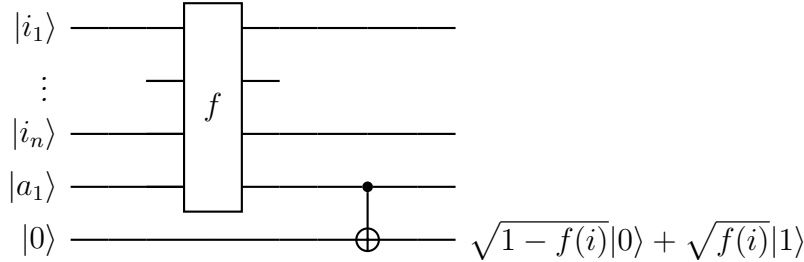


FIGURE 19. Circuit diagram for Q_f where $f : \{0, 1\}^n \rightarrow \{0, 1\}$.

General integrals can always be split into R integration problems for precision 2^{-R} , i.e.,

$$\frac{1}{N} \sum_{i=0}^{N-1} f(i) = \frac{1}{N} \sum_{i=0}^{N-1} \sum_{k=1}^R 2^{-k} \underbrace{f_k(i)}_{\in\{0,1\}}.$$

5.2. Solving linear systems of equations. We now consider the question of (approximate) solution of (large) linear systems of equations, i.e., for $A \in \mathbb{C}^{N \times N}$ Hermitian and invertible and $b \in \mathbb{C}^N$, the goal is to find $x \in \mathbb{C}^N$ with $Ax = b$.

The quantum version of this reads, due to normalizations, slightly different.

Problem formulation. Let $A \in \mathbb{C}^{N \times N}$ be Hermitian with $\det A = 1$, $(b_i) = b \in \mathbb{C}^N$, and $(x_i) = x \in \mathbb{C}^N$ such that $Ax = b$ for some $N = 2^n$. Furthermore, let $|b\rangle$ be an n -qubit state given by

$$|b\rangle := \frac{\sum_i b_i |i\rangle}{\|\sum_i b_i |i\rangle\|} \quad (6)$$

and

$$|x\rangle := \frac{\sum_i x_i |i\rangle}{\|\sum_i x_i |i\rangle\|}. \quad (7)$$

The goal is to find a quantum state $|\tilde{x}\rangle$ with $\| |x\rangle - |\tilde{x}\rangle \| \leq \varepsilon$ with probability $\Omega \geq 1/2$, for a given error $\varepsilon > 0$. This is sometimes (formal!) written as $A|x\rangle = |b\rangle$.

Remark 49.

- *Normalization in the formulation is necessary to get a quantum state. Moreover, the solution to the quantum linear system is a quantum state, where the amplitudes encode information of the solution of the classical linear system.*

- *If A is not Hermitian, one can consider the Hermitian matrix*

$$\tilde{A} = \begin{pmatrix} 0 & A^H \\ A & 0 \end{pmatrix} = |1\rangle\langle 0| \otimes A + |0\rangle\langle 1| \otimes A^H.$$

As we have doubled the dimension of the problem, we need 1 additional ancilla qubit. We then solve $\tilde{A}|0x\rangle = |1b\rangle$.

- *If $\det A \neq 1$, A can be rescaled such that $\det A = 1$.*

Remark 50.

- *We want an efficient algorithm for the quantum linear system of equations (gate complexity poly-logarithmic in N).*

- *We implicitly assume that the state $|b\rangle$ can be provided in an efficient way and that A can be implemented on a quantum computer.*
- *Extracting amplitudes from the state $|x\rangle$ has complexity $\mathcal{O}(N)$. Therefore, using a quantum linear system is only useful if quantum states are required, e.g. as subroutine in another quantum algorithm.*

Classical algorithm. In order to compare potential speed-ups of a quantum algorithm, we compare with the most commonly used iterative solver, the conjugate gradient (CG) method.

If A is positive definite and sparse with sparsity s , then the CG method needs $\mathcal{O}(sN\ell)$ operations, where ℓ is the number of iterations of the CG method.

Using the condition number $\kappa(A) = \|A\| \|A^{-1}\|$, the error between exact solution x and ℓ -th iteration of the CG method in the A -norm is controlled by

$$\|x - x_\ell\|_A \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^\ell \|x - x_0\|_A.$$

Let $\varepsilon > 0$ be given such that

$$2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^\ell \leq \varepsilon,$$

i.e., we obtain a bound for the relative error $\frac{\|x - x_\ell\|_A}{\|x - x_0\|_A} \leq \varepsilon$. Defining

$$q := \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} = 1 - \frac{2}{\sqrt{\kappa} + 1} \geq \exp\left(-\frac{2}{\sqrt{\kappa}}\right),$$

it follows that

$$\ell \leq \frac{\log \frac{\varepsilon}{2}}{\log q} \leq \frac{1}{2} \sqrt{\kappa} \ln \frac{2}{\varepsilon}.$$

The computational complexity of the CG method is therefore $\mathcal{O}(sN\sqrt{\kappa} \ln(2/\varepsilon))$ and the aim of the following is to improve on that with a quantum algorithm.

Quantum algorithm (HHL). In the following, we introduce the HHL (Harrow-Hassidim-Lloyd) quantum algorithm for solution of quantum linear systems.

Idea: Given a Hermitian matrix $A \in \mathbb{C}^{N \times N}$, we use the spectral decomposition

$$A = \sum_{j=0}^{N-1} \lambda_j |u_j\rangle\langle u_j|$$

with the eigenvalues $\lambda_j \in \mathbb{R}$ and eigenvectors $|u_j\rangle \in \mathbb{R}^N$ of A . Now, as

$$|b\rangle = \sum_{j=0}^{N-1} \beta_j |u_j\rangle = \sum_{j=0}^{N-1} \langle u_j | b \rangle |u_j\rangle$$

it follows from the spectral theorem that

$$A^{-1} = \sum_{j=0}^{N-1} \frac{1}{\lambda_j} |u_j\rangle\langle u_j|$$

and

$$|x\rangle = A^{-1}|b\rangle = \sum_{j=0}^{N-1} \frac{\beta_j}{\lambda_j} |u_j\rangle.$$

Note that this should be seen as formal calculation, the correct quantum operations are shown later.

However, the matrices A, A^{-1} are, in general, not unitary and hence the application of A^{-1} as matrix-vector-multiplication $A^{-1}|b\rangle$ is not allowed!

Solution: The matrix $U = \exp(iA)$ is unitary and has the same eigenvectors as $A = XDX^{-1}$ since

$$e^{iA} = \sum_{n=0}^{\infty} \frac{(iA)^n}{n!} = \sum_{n=0}^{\infty} \frac{(iXDX^{-1})^n}{n!} = X \sum_{n=0}^{\infty} \frac{(iD)^n}{n!} X^{-1} = X e^{iD} X^{-1} = \sum_{j=0}^{N-1} e^{i\lambda_j} |u_j\rangle\langle u_j|.$$

We can therefore proceed as follows:

- 1) compute eigenvalues of U with quantum phase estimation, which are (up to the exponential $e^{i\bullet}$) the same as the eigenvalues of A .
- 2) Invert the eigenvalues of U with a *controlled rotation*, ($\lambda_j \mapsto 1/\lambda_j$) and
- 3) reverse quantum phase estimation.

5.3. Hamiltonian simulation. Implementing $\exp(iAt) = U$ directly has computational complexity $\mathcal{O}(N^3)$ and is therefore not favorable. We therefore have to answer the question on how we can find an efficient implementation of $\exp(iAt) = U$ and U^{2^k} ?

For a Hamiltonian of U , this is a fundamental problem in quantum computing. Schrödinger's equation

$$i \frac{d}{dt} |\psi(t)\rangle = H |\psi(t)\rangle \quad (8)$$

describes every q-system. Furthermore, the solution to eq. (8) is given by

$$|\psi(t)\rangle = e^{-iHt} |\psi(0)\rangle.$$

Thus, for an efficient simulation (*Hamiltonian simulation*), we have to implement e^{-iHt} in an efficient way. This can be done as a circuit up to an error ε .

Definition 51. A Hamiltonian H acting on n qubits can be implemented efficiently if

$$\forall t > 0, \varepsilon > 0 \exists \text{ a quantum circuit } U_H \text{ such that } \|U_H - \exp(iHt)\| \leq \varepsilon$$

where U_H consists of $\text{poly}(n, t, 1/\varepsilon)$ gates.

Remark 52. In general the minimal time required to simulate H at time t is $\mathcal{O}(\|H\|t)$ (no fast forwarding theorem).

For the general approximation problem, it is NP-hard to find a gate decomposition. Hence we will make assumptions to reduce the problems complexity.

5.3.1. k -local Hamiltonians. Assume that

$$H = \sum_{j=0}^{N-1} H_j, \quad m \sim \text{poly}(n)$$

and that H_j acts on $k = \mathcal{O}(1)$ qubits.

5.3.2. Trotter-Suzuki splitting. The implementation of $\exp(iH_j t)$ is easier for k -local Hamiltonians, as each H_j acts on k qubits. If H is diagonalizable ($H = TDT^H$), we can write $\exp(iHt)$ as

$$\exp(iHt) = T \exp(iDt) T^H.$$

If D_{ll} can be determined efficiently, we can load the entry (i), apply the phase (ii), and unload the entry (iii) to obtain

$$|i0\rangle \xrightarrow{(i)} |iD_{ll}\rangle \xrightarrow{(ii)} e^{iD_{ll}t} |iD_{ll}\rangle \xrightarrow{(iii)} e^{iD_{ll}t} |i0\rangle.$$

We have thus found an efficient diagonalization of H_j for k -local Hamiltonians.

Remark 53. In general

$$e^{iHt} \neq \prod_{j=1}^m e^{iH_j t},$$

as the equality only holds if all H_j commute as

$$e^{A+B} = \sum_{n=0}^{\infty} \frac{(A+B)^n}{n!} = \sum_{n=0}^{\infty} \sum_{k=0}^n \binom{n}{k} \frac{A^k B^{n-k}}{n!} \stackrel{\text{Cauchy}}{=} \sum_{n=0}^{\infty} \frac{A^n}{n!} \sum_{k=0}^n \frac{B^k}{k!} = e^A e^B$$

where we have used

$$\binom{n}{k} \frac{1}{n!} = \frac{1}{k!(n-k)!}.$$

Theorem 54. *Trotter/Lie-product formula*

Let $H = H_1 + H_2$ with H, H_1, H_2 Hermitian. Then

$$\lim_{l \rightarrow \infty} (e^{iH_1 t/l} e^{iH_2 t/l})^l = e^{iHt}$$

and

$$\|e^{iHt} - (e^{iH_1 t/L} e^{iH_2 t/L})^L\| \leq c \frac{t^2}{L}$$

with $c = c(\|H_1\|, \|H_2\|)$.

Proof. We start with a Taylor expansion of $\exp(iH_1 t/L)$:

$$e^{iH_1 t/L} = \text{id} + iH_1 \frac{t}{L} + \underbrace{\mathcal{O}(\|H_1\|^2 \frac{t^2}{L^2})}_{\text{bounded terms in } H_1, t, L}$$

which implies

$$e^{iH_1 t/L} - e^{iH_2 t/L} = \left(\text{id} + iH_1 \frac{t}{L} + \mathcal{O}\left(\|H_1\|^2 \frac{t^2}{L^2}\right) \right)^2 \left(\text{id} + iH_2 \frac{t}{L} + \mathcal{O}\left(\|H_2\|^2 \frac{t^2}{L^2}\right) \right)^2 \quad (9)$$

$$= \left(\underbrace{\text{id} + i(H_1 + H_2) \frac{t}{L}}_{=:A} + \underbrace{\mathcal{O}\left(\max(\|H_1\|, \|H_2\|)^2 \frac{t^2}{L^2}\right)}_{=:B} \right)^2. \quad (10)$$

Now we can use the binomial theorem to obtain

$$(A + B)^2 = A^2 + \sum_{j=0}^{L-1} A^{L-j-1} B A^j + \underbrace{\dots + B^2}_{\mathcal{O}(\|B\|^2)}$$

and hence eq. (10) reads

$$\begin{aligned} & e^{i(H_1+H_2)t} + L \mathcal{O}\left(\max(\|H_1\|, \|H_2\|)^2 \frac{t^2}{L^2}\right) \\ \implies & \|e^{i(H_1+H_2)t} - (e^{iH_1 t/L} e^{iH_2 t/L})^L\| \leq c \frac{t^2}{L}. \end{aligned}$$

Finally, we have

$$\lim_{L \rightarrow \infty} c \frac{t^2}{L} \rightarrow 0.$$

□

Remark 55.

- For efficient simulation $\max(\|H_1\|, \|H_2\|) = \mathcal{O}(\text{poly}(n))$.
- Given a maximal permissible error $\leq \varepsilon$, L is given by $L = \mathcal{O}(c(H_1, H_2)t^2/\varepsilon)$.

- In this case we use first order splitting. However, higher order splitting is also possible, e.g., Strang splitting (second order):

$$\|e^{i(H)t} - (e^{i(H_2)t/(2L)}e^{i(H_1)t/L}e^{i(H_2)t/(2L)})^L\| \leq c \frac{t^3}{L^2}$$

and thus

$$L = \mathcal{O}\left(c(H_1, H_2)t^{\frac{3}{2}}/\sqrt{\varepsilon}\right).$$

Splitting methods of order p are possible with

$$L = \mathcal{O}\left(t^{\frac{p+1}{p}}\varepsilon^{-\frac{1}{p}}\right)$$

which is almost linear in t .

- If k -local Hamiltonians with m -terms are used we obtain

$$\|e^{i(H)t} - (e^{i(H_1)t/L} \dots e^{i(H_m)t/L})^L\| \leq c \frac{m^2 t^2}{L}.$$

- In practice, we tend to over estimate the error. However, the advantages of this method are that it is simple and does not need additional qubits.

5.4. Graph coloring method. An essential question that arises is how can we find the decomposition

$$H = \sum_i H_i.$$

One possible answer is using *graph coloring methods*.

Definition 56. An undirected graph $G(V, E)$ is defined by the set of vertices V and the set of edges $E \subseteq V \times V$ which are unordered pairs of vertices.

Example 57. Let $V = \{1, \dots, 7\}$ and $E = \{(1, 2), (2, 3), (2, 6), (2, 7), (6, 7), (3, 4), (4, 5)\}$. The graph $G(V, E)$ is shown in fig. 20.

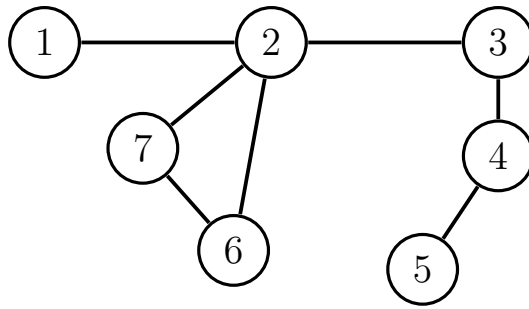


FIGURE 20. Graph corresponding to V and E in theorem 57

Definition 58. Two vertices $v_1, v_2 \in V$ are called adjacent if $(v_1, v_2) \in E$. The degree of a vertex $v \in V$ is the number of adjacent vertices $\#\{v_i : (v, v_i) \in E\}$.

Let $|v| = n$. A graph G can be represented by its $n \times n$ adjacency matrix A given by

$$A_{ij} = \begin{cases} 1 & \text{if } v_i, v_j \in V : (v_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases}.$$

Example 59. The adjacency matrix of theorem 57 is given by

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

5.4.1. *Graph coloring problem.* Is it possible to color the edges of a graph such that no two adjacent edges have the same color, using k different colors?

5.4.2. *Solution.* The answer to this problem is given by the following theorem.

Theorem 60. *Vizing's theorem* For any graph $G = (V, E)$ with maximum degree d , the graph can be colored according to section 5.4.1 with $k \leq d + 1$ colors.

Usage for Hamiltonian simulation:

- 1) Identify H with adjacency matrix A such that

$$A_{ij} = \begin{cases} 1 & \text{if } H_{ij} \neq 0 \\ 0 & \text{if } H_{ij} = 0. \end{cases}$$

We also obtain the associated graph G .

- 2) Find a coloring of G with k colors.
- 3) Split G (A) into k sub-graphs based on color

$$A = \sum_{c=1}^k A_c.$$

For arbitrary matrices A this is impossible to implement efficiently. We therefore require A to be sparse.

5.4.3. *Assumptions.* Let $A \in \mathbb{C}^{2^n \times 2^n}$ be s -sparse and let us have *sparse access* to entries of A , namely

- each row/column of A has at most s non-zero entries.
- We have the *query*

$$O_A : |ij\rangle|0\rangle \mapsto |ij\rangle|A_{ij}\rangle$$

where the back register is large enough to store the entries $A_{ij} \in \mathbb{C}$ exact (or with high enough precision) in binary representation.

- We have another Query

$$O_C : |jl\rangle \mapsto |j\nu(j, l)\rangle$$

where $\nu(j, l) \in \{1, \dots, N-1\}$ is the position of the l -th non-zero entry in the j -th row of A .

- We can execute O_A^{-1} and O_C^{-1} .

Remark 61. *Summary*

- With s -sparsity, the graph G has maximum degree $d \leq s$.
- Vizing's theorem states that G can be colored with at most $s + 1$ colors.
- A coloring with s^2 colors can be computed efficiently.

- The A_c matrices are symmetric and 1-sparse. We can therefore efficiently simulate H_C since the matrices are diagonalizable with $\mathcal{O}(1)$ as w.o.l.g.:

$$H = \underbrace{\text{diag}(H)}_{\text{efficient}} + \text{remainder}.$$

- Total computational complexity for a naive implementation is

$$\mathcal{O}(s^2 t^2 \text{poly}(n)/\varepsilon).$$

5.4.4. *Sparse access implementation.* How can we implement *sparse access*?

Example 62. *Circulant matrix* Consider the 3-sparse circulant matrix A given by

$$A := \begin{pmatrix} \alpha & \gamma & 0 & \cdots & 0 & \beta \\ \beta & \alpha & \gamma & 0 & & 0 \\ 0 & \cdots & \cdots & \cdots & & \vdots \\ \vdots & & 0 & \beta & \alpha & \gamma \\ \gamma & & 0 & 0 & \beta & \alpha \end{pmatrix}.$$

First, ν is given by $\nu(j, l) = j + l - 1 \bmod N$, for $j = 0, 1, 2$. Note that the matrix indices start with 0. Next, O_C is given by

$$O_C : |j\rangle \mapsto \begin{cases} |\text{mod}(j-1, N)\rangle & l = 0 \\ |j\rangle & l = 1 \\ |\text{mod}(j+1, N)\rangle & l = 2 \end{cases}$$

where the $\text{mod}(j \pm 1, N)$ operation can be implemented using shift permutations.

Before we can implement the circuits required to implement O_A and O_C we consider the matrices

$$R = \begin{pmatrix} 0 & 1 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & & & 1 \\ 1 & 0 & \cdots & 0 \end{pmatrix}, \quad L = \begin{pmatrix} 0 & \cdots & 0 & 1 \\ 1 & & & 0 \\ \vdots & \ddots & & \vdots \\ 0 & \cdots & 1 & 0 \end{pmatrix}.$$

L circuit:

For 3 qubits as in fig. 21 the bitflip X is given by

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

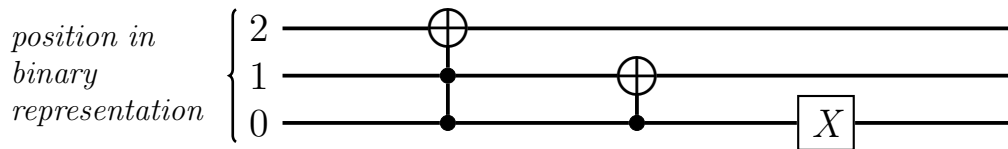


FIGURE 21. Circuit diagram for **L**

Example 63. For $2 \mapsto 3$ we have,

$$\underbrace{|0\rangle}_2 \underbrace{|1\rangle}_1 \underbrace{|0\rangle}_0 \mapsto \underbrace{|0\rangle}_2 \underbrace{|1\rangle}_1 \underbrace{|1\rangle}_0$$

and for $7 \mapsto 0$

$$|111\rangle \mapsto |000\rangle.$$

\boxed{R} circuit:

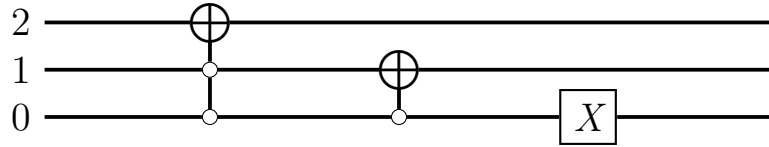
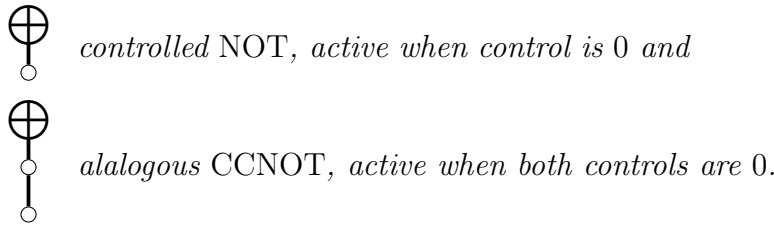


FIGURE 22. Circuit diagram for \boxed{R}

where



We also need $\boxed{L^2}$ and $\boxed{R^2}$: For L^2 there holds

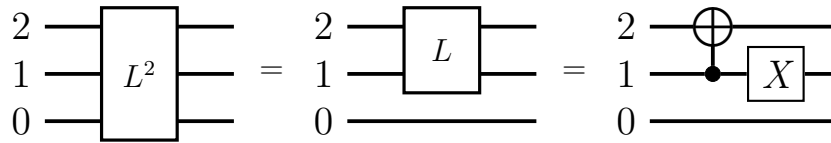


FIGURE 23. Circuit diagram for $\boxed{L^2}$

and the analogue is true for R .

O_C circuit:

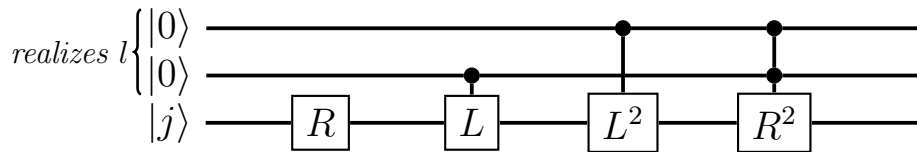


FIGURE 24. Circuit diagram for O_C

Depending on l we have the following cases:

- $l = 0$: R -gate for all l , $j \mapsto \text{mod}(j - 1, N)$. This is OK for $l = 0$, hence $|00\rangle$.
- $l = 1$: For $l = 1$, $j \mapsto j$ and we obtain $|01\rangle$. Here we can use the L -gate to reverse the R -gate.
- $l = 2$: For $l = 2$, $j \mapsto \text{mod}(j + 1, N)$ and we obtain $|10\rangle$. We can use an L^2 -shift since RL^2 results in an L -shift.
- $l = 3$: $l = 3 \simeq |11\rangle$ represents the 0 entries in the matrix. We can use the R^2 -shift to reverse the L^2 -shift.

O_A circuit:

In general, controlled rotations are unitary operations. Let $|\Theta\rangle$ be the control and $|0\rangle$ be the target. We define

$$U_\Theta : |\Theta\rangle|0\rangle \mapsto |\Theta\rangle(\cos(\pi\Theta)|0\rangle + \sin(\pi\Theta)|1\rangle)$$

where $\Theta \in [0, 1]$. Note that the binary representation of Θ is given by $\Theta = \Theta_0 2^{-1} + \Theta_1 2^{-2} + \dots + \Theta_{d-1} 2^{-d}$.

For $|\Theta\rangle = |0\rangle$ we obtain the output $|0\rangle|0\rangle$. For $|\Theta\rangle = |1\rangle$ we obtain a rotation of $\pi\Theta$ around the y -axis.

In matrix form, we have

$$R(2J) := \begin{pmatrix} \cos(J) & -\sin(J) \\ \sin(J) & \cos(J) \end{pmatrix}$$

which represents a 1-qubit rotation around the y -axis. We will write $\boxed{R(J)}$ for the controlled rotation in circuit diagrams.

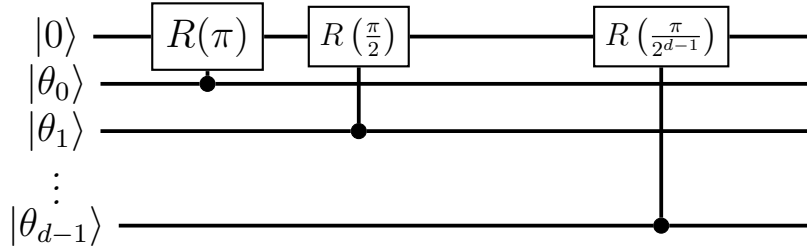


FIGURE 25. Circuit diagram for O_A

Finally, we generate the value $\alpha \in [0, 1]$ by controlled rotation of $\Theta_1 = 1/\pi \arccos \alpha$. Similarly, we generate β and γ as $\Theta_2 = 1/\pi \arccos \beta$ and $\Theta_3 = 1/\pi \arccos \gamma$ respectively.

5.5. Modification for 3-diagonal matrices. We will add two additional rotations $R(\Theta_4)$ and $R(\Theta_5)$ which rotate A_{1n} (A_{n1}) back to 0. We thus need an additional control qubit.

5.6. Loading the right hand side. Let $b \in \mathbb{R}^2$ be arbitrary but fixed. We want to load the state $|b\rangle := \sum_i b_i |i\rangle / \|b\|$ in $\mathcal{O}(\log N)$ (or $|\tilde{b}\rangle \simeq |b\rangle$ up to machine precision). A possible solution is *quantum RAM* (qRAM), e.g., a classical data structure which can be accessed with superpositions on q-states.

Let, without loss of generality, $\|b\| = 1$ which can be loaded in binary representation as it is a fixed scalar.

5.6.1. Idea. The idea we want to use is divide and conquer. We start by saving x in a binary tree B_x with root $\sum_{i=1}^N x_i^2 = 1$, depth $\log N = n$, and leafes $(x_i^2, \text{sgn } x_i)$ as shown in fig. 26.

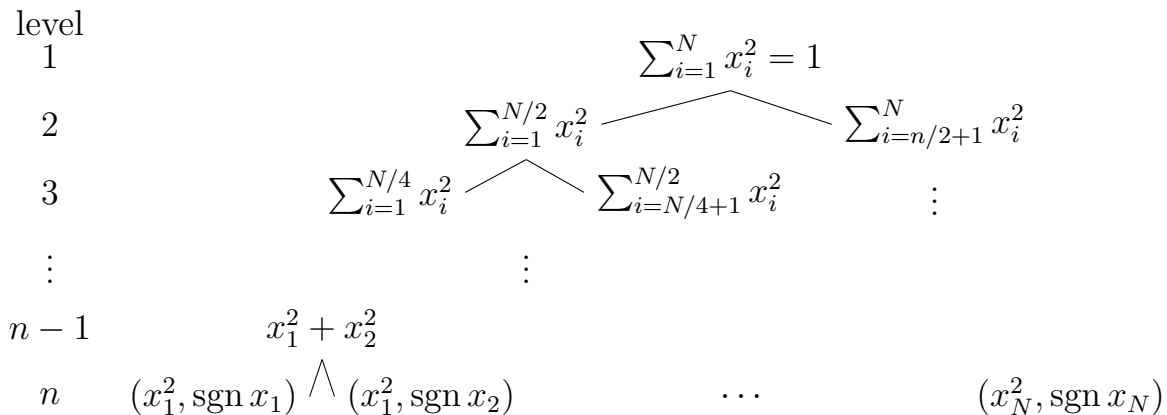


FIGURE 26. Divide and conquer tree for $\sum_{i=1}^N x_i^2 = 1$

- Remark 64.**
- Each node is the sum of its children. Additionally, each leaf has a sign bit.
 - To load a vector, traverse the tree, starting from the root and add new registers if necessary. Controlled rotation is used to realize the value in a node.
 - B_x has $\mathcal{O}(N)$ nodes. We assume that the node values are precomputed (classical) and are therefore not included in the complexity.

Lemma 65. If $x \in \mathbb{R}^N$ is precomputed in B_x , then algorithm 6 generates the state

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

in $\mathcal{O}(n) = \mathcal{O}(\log N)$.

Proof. First, we show that algorithm 6 is correct and actually generates the state $|x\rangle$. Go through the tree, multiplying nodes u_k with $\sqrt{u_k/u_{k-1}}$ (for leaf i add an additional $\text{sgn}(x_i)$), where k is the current level. This way, we obtain

$$\prod_{k=1}^n \sqrt{u_k/u_{k-1}} \text{sgn}(x_i) = \underbrace{\sqrt{\frac{u_n}{u_1}}}_{\frac{x_i^2}{1}} \text{sgn}(x_i) = x_i.$$

It remains to show the complexity. There are 2^k rotations (in $\mathcal{O}(1)$) for each level k . This happens in parallel. Thus, a controlled operation on the same qubits. The control checks the binary register of the parent. The complexity is therefore bounded with

$$\#\text{levels} = n = \log N.$$

□

Remark 66. The assumption that B_x is precomputed is significant and can ruin the speedup. Nonetheless, if B_x is precomputed, theorem 65 shows that $|b\rangle$ and copies of $|b\rangle$ can be generated in $\mathcal{O}(\log N)$.

Algorithm 6 (Loading a vector from qRAM)

```

1: input:  $x \in \mathbb{R}^N$  represented by its binary tree  $B_x$ 
2: output:  $|x\rangle$ 
3: initialize  $n$ -qubits  $|0\dots 0\rangle$  with  $|q_1\rangle, \dots, |q_n\rangle$ .
4:  $r = \text{root}(B_x)$ , call  $\text{PROCESSNODE}(r)$ 
5: procedure  $\text{PROCESSNODE}(\text{vertex } u)$ 
6:    $u_l, u_r$  are the children of  $u$ 
7:    $\theta_u = \arccos(\sqrt{u_l/u})$ 
8:   controlled rotation  $|q_k\rangle = \cos\theta|0\rangle + \sin\theta|1\rangle$ , if qubits  $|q_1\rangle, \dots, |q_k\rangle \simeq$  binary representation of  $u$ 
9:   if  $u_l, u_r$  are leafs then
10:     $|q_k\rangle = \text{PROCESSSIGN}(q_k, u_l, u_r)$ 
11:    return
12:   else
13:     $\text{PROCESSNODE}(u_l)$ 
14:     $\text{PROCESSNODE}(u_r)$ 
15:   end if
16: end procedure
17: procedure  $\text{PROCESSSIGN}(q_k, u_l, u_r)$ 
18:   if  $\text{sgn}(u_l) = \text{sgn}(u_r) = 1$  then
19:    return  $q_k$ 
20:   else if  $\text{sgn}(u_l) = \text{sgn}(u_r) = -1$  then
21:    return  $-q_k$ 
22:   else if  $\text{sgn}(u_l) = 1, \text{sgn}(u_r) = -1$  then
23:    return  $Zq_k$ 
24:   else if  $\text{sgn}(u_l) = -1, \text{sgn}(u_r) = 1$  then
25:    return  $-Zq_k$ 
26:   end if
27: end procedure

```

5.7. **HHL revisited.** Let us now revisit and improve the HHL algorithm. Consider the circuit shown in fig. 27.

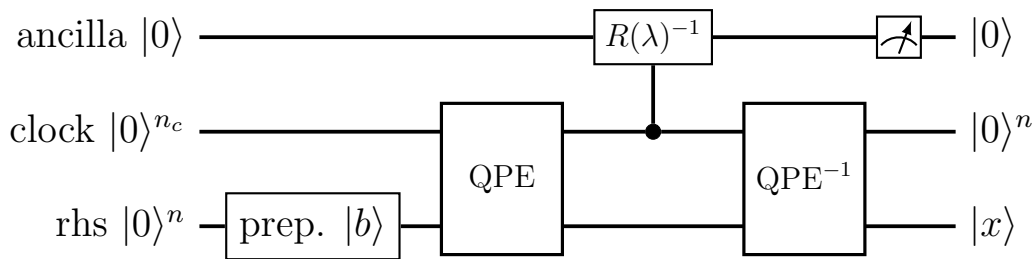


FIGURE 27. Revisited circuit for HHL

- 1) Determine $|b\rangle$ (e.g. qRAM) with $n = \log N$ qubits to represent $|b\rangle$
- 2) Apply quantum phase estimation to

$$|0\rangle|b\rangle = \sum_{j=0}^{N-1} \beta_j |0\rangle|u_j\rangle$$

where $U = \exp(iA)$ and $|0\rangle = |0^n\rangle$. We therefore need $H^{\otimes n_c}$, U^{2^j} for $j = 0, \dots, n_c - 1$ and QFT with $\mathcal{O}(n^2)$ gates to obtain an approximation for the eigenvalues of A (not U). The result is an approximation as the quantum phase estimation assumes λ_j has an exact binary representation in n_c qubits. The implementation of QPE as circuit is given by fig. 28

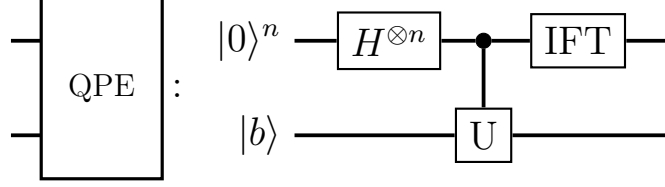


FIGURE 28. QPE circuit diagram

where

$$\begin{array}{|c|} \hline \text{U} \\ \hline \end{array} \text{ implements } |j\rangle|psi\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{j\lambda_i} |j\rangle|\psi\rangle \text{ with} \\ \text{eigenvalues } \lambda \text{ and corresponding eigenvectors } \psi.$$

We apply QPE to $|b\rangle$, i.e.,

$$|0\rangle|b\rangle \mapsto \sum_j \beta_j |\tilde{\lambda}_j\rangle |u_j\rangle$$

where $\tilde{\lambda}_j$ is the binary representation of λ_j .

- 3) Next, we add an ancilla qubit to use controlled rotation ($R_y(2J)$) to rotate $\theta/2 = \arcsin(c/\tilde{\lambda}_j)$ around the y -axis. We obtain

$$\sum_j \beta_j |\tilde{\lambda}_j\rangle |u_j\rangle \left(\sqrt{1 - \frac{c^2}{\tilde{\lambda}_j^2}} |0\rangle + \frac{c}{\tilde{\lambda}_j} |1\rangle \right)$$

where c is a constant such that $c \leq \min_j |\lambda_j| = \mathcal{O}(1/k)$.

Note that $\arcsin(\alpha)$ can be realized (approximated) with $\mathcal{O}(\text{poly}(n))$ elementary gates.

- 4) Apply the inverse QPE (*uncomputing*) to obtain

$$\sum_{j=0}^{N-1} \beta_j |0\rangle |u_j\rangle \left(\sqrt{1 - \frac{c^2}{\tilde{\lambda}_j^2}} |0\rangle + \frac{c}{\tilde{\lambda}_j} |1\rangle \right).$$

- 5) Measure the last qubit. If the result is $|1\rangle$ the state

$$c \sum_{j=0}^{N-1} \frac{\beta_j}{\tilde{\lambda}_j} |0\rangle |u_j\rangle$$

is proportional to $|\tilde{x}\rangle$.

The probability to measure $|1\rangle$ is

$$\frac{1}{\sqrt{\sum_{j=0}^{N-1} \frac{c^2 |\beta_j|^2}{|\tilde{\lambda}_j|^2}}}.$$

Remark 67. *The normalization factor cancels with c . It is therefore not present in the solution but in the probability of success.*

Example 68. *Quantum composer* Let us consider the 1-qubit system:

$$A = \begin{pmatrix} 1 & -\frac{1}{3} \\ -\frac{1}{3} & 1 \end{pmatrix}, \quad b = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

with the solution to $Ax = b$ being

$$x = \frac{1}{8} \begin{pmatrix} 3 \\ 9 \end{pmatrix}.$$

The eigenvalues $\lambda_{1,2}$ and eigenvectors $u_{1,2}$ of A are given by

$$\lambda_0 = \frac{2}{3}, \quad \lambda_1 = \frac{4}{3}, \quad u_0 = \frac{1}{\sqrt{2}} \begin{pmatrix} -1 \\ -1 \end{pmatrix}, \quad u_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} -1 \\ 1 \end{pmatrix}.$$

We now need $U = \exp(iAt)$ and $U^2 = \exp(i2At)$. Choosing $t = 3\pi/4$ the eigenvalues can be represented with 2 qubits in QPE. For this example, U is given by

$$U = \frac{1}{\sqrt{2}} \begin{pmatrix} -1 & -1 \\ -1 & 1 \end{pmatrix} \underbrace{\begin{pmatrix} e^{i\lambda_0 t} & 0 \\ 0 & e^{i\lambda_1 t} \end{pmatrix}}_{\begin{pmatrix} i & 0 \\ 0 & -1 \end{pmatrix}} \frac{1}{\sqrt{2}} \begin{pmatrix} -1 & -1 \\ -1 & 1 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} -1+i & 1+i \\ 1+i & -1+i \end{pmatrix}$$

and similarly

$$U^2 = \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix}.$$

For the implementation with a 4-parameter unitary gate in IBM-Q given by

$$U = \begin{pmatrix} e^{i\gamma} \cos(\theta/2) & -e^{i(\gamma+\lambda)} \sin(\theta/2) \\ e^{i(\gamma+\phi)} \sin(\theta/2) & e^{i\gamma+\phi+\lambda} \cos(\theta/2) \end{pmatrix}$$

we obtain

$$\theta = \frac{\pi}{2}, \quad \phi = -\frac{\pi}{2}, \quad \lambda = \frac{\pi}{2}, \quad \gamma = \frac{3\pi}{4}$$

for U and

$$\theta = \pi, \quad \phi = \pi, \quad \lambda = 0, \quad \gamma = 0$$

for U^2 .

controlled rotation:

The eigenvalues from QPE (2-qubits) are $\tilde{\lambda}_j = N\lambda_j t/2\pi$ and hence

$$\tilde{\lambda}_0 = 1, \quad \tilde{\lambda}_1 = 2.$$

Choosing $c = 1$ we obtain

$$\theta_1 = 2 \arcsin \left(\frac{1}{\tilde{\lambda}_1} \right) = \pi,$$

$$\theta_1 = 2 \arcsin \left(\frac{1}{\tilde{\lambda}_2} \right) = \frac{\pi}{3}.$$

results:

The output are normalized probability values

state	P
$ 00\rangle$	0.1875
$ 01\rangle$	0.0625
$ 10\rangle$	0.1875
$ 11\rangle$	0.5625

(11)

where the first qubit is $|b\rangle$ and the second qubit is the ancilla qubit. In this case, only the states where the ancilla qubit is $|1\rangle$ are relevant, i.e., $|01\rangle$ and $|11\rangle$. For this conditional measurement of the ancilla qubit we obtain $|0\rangle$ with $p = 1/10$ and $|1\rangle$ with $p = 9/10$. The normalized solution is thus given by

$$|x\rangle = \frac{3}{\sqrt{90}}|0\rangle + \frac{9}{\sqrt{90}}|1\rangle.$$

5.8. Error and complexity analysis. Until now we assumed that all scalars have an exact binary representation. However, in general this is not feasible for all eigenvalues of A . Quantum phase estimation outputs eigenvalues $\tilde{\lambda}_j$ which are approximations of λ_j with error

$$|\tilde{\lambda}_j \lambda_j| \leq \delta$$

with probability $1 - 1/\text{poly}(n)$. The run time of QPE is $\mathcal{O}(T_u \text{poly}(n)/\delta)$ where T_u is the time required to implement $U = \exp(iA)$.

5.8.1. problem. The next problem is the numerical stability of

$$\lambda_j \mapsto \frac{1}{\lambda_j}$$

with controlled rotation. If $\lambda_j \approx 0$ a small error $\tilde{\lambda}_j + \varepsilon$ has a large impact on the result. The effect of the small perturbation ε is significantly larger than ε itself as

$$\left| \frac{1}{\lambda_j} - \frac{1}{\tilde{\lambda}_j} \right| = \left| \frac{\varepsilon}{\lambda_j(\lambda_j + \varepsilon)} \right| \approx \frac{\varepsilon}{\lambda_j^2}.$$

The problem has a bad condition number $\kappa(A)$.

5.8.2. solution. The solution are filter functions. We only invert eigenvalues λ if $\lambda \geq 1/\kappa$. Accordingly, we define f as

$$f(\lambda) := \begin{cases} 0 & \text{for } \lambda < \frac{1}{2\kappa} \\ \frac{1}{2\kappa\lambda} & \text{for } \lambda \geq \frac{1}{\kappa} \\ \frac{1}{2} \sin\left(\frac{\pi}{2}(2\kappa\lambda - 1)\right) & \text{otherwise.} \end{cases}$$

Note that for $1/(2\kappa) \leq \lambda < 1/\kappa$, f is interpolating between 0 and $1/(2\kappa\lambda)$ and is thus continuous.

Similarly, we define the reverse filter g as

$$g(\lambda) := \begin{cases} 0 & \text{for } \lambda \geq \frac{1}{\kappa} \\ \frac{1}{2\kappa\lambda} & \text{for } \lambda < \frac{1}{2\kappa} \\ \frac{1}{2} \cos\left(\frac{\pi}{2}(2\kappa\lambda - 1)\right) & \text{otherwise.} \end{cases}$$

Note that g is also continuous and $f^2(\lambda) + g^2(\lambda) \leq 1$ for all λ .

Remark 69. *The choice of f and g is not unique, other options are possible. The cut-off $1/(2\kappa)$ can also be chosen differently.*

Instead of controlled rotation, add a 3-qubit register:

$$|h(\tilde{\lambda}_j)\rangle := \sqrt{1 - f(\tilde{\lambda}_j)^2 - g(\tilde{\lambda}_j)^2}|\text{nothing}\rangle + f(\tilde{\lambda}_j)|\text{well}\rangle + g(\tilde{\lambda}_j)|\text{ill}\rangle$$

where $|\text{nothing}\rangle$ is the part where no inversion was performed, $|\text{well}\rangle$ is the part where the eigenvalues have been inverted and $|\text{ill}\rangle$ is the part of $|b\rangle$ inside the ill-conditioned subspace of A .

Finally, we apply $(QPE)^{-1}$ and measure $|\text{well}\rangle$.

The result is roughly

$$\sum_{j, \lambda_j \geq 1/\kappa} \lambda_j^{-1} \beta_j |u_j\rangle |\text{well}\rangle + \sum_{j, \lambda_j < 1/\kappa} \beta_j |u_j\rangle |\text{ill}\rangle.$$

Lemma 70. *The map $\lambda \mapsto |h(\lambda)\rangle$ is Lipschitz continuous with Lipschitz constant $L = \mathcal{O}(\kappa)$, namely*

$$\| |h(\lambda_i)\rangle - |h(\lambda_j)\rangle \|_2 \leq c\kappa |\lambda_i - \lambda_j|. \quad (12)$$

Proof. After estimating the derivatives of f and g , eq. (12) can be verified explicitly. \square

5.8.3. *Goal.* Our next Goal is an error estimation for the inexact QPE with filtered inversion of the eigenvalues. The exact state and the approximated state are given by

$$\begin{aligned} \text{exact:} & \quad |\Psi\rangle := \sum_i \beta_i |u_i\rangle |h(\lambda_i)\rangle \\ \text{approximation:} & \quad |\tilde{\Psi}\rangle := \sum_i \beta_i |u_i\rangle |h(\tilde{\lambda}_i)\rangle. \end{aligned}$$

Hence the error in the 2-norm is

$$\| |\Psi\rangle - |\tilde{\Psi}\rangle \|_2^2 = \| |\Psi\rangle \|_2^2 + \| |\tilde{\Psi}\rangle \|_2^2 - 2 \left(1 - \underbrace{\Re\langle \Psi | \tilde{\Psi} \rangle}_{\in[0,1], \text{ (C.S.)}} \right). \quad (13)$$

For u_i ONB there holds

$$\Re\langle \Psi | \tilde{\Psi} \rangle = \sum_{i=1}^N |\beta_i|^2 \langle h(\lambda_i) | h(\tilde{\lambda}_i) \rangle$$

and using ??

$$\Re\langle \Psi | \tilde{\Psi} \rangle \geq 1 - \frac{c^2 \kappa^2}{2} |\lambda_i - \tilde{\lambda}_i|^2 \geq 1 - \frac{c^2 \kappa^2 \delta^2}{2}. \quad (14)$$

Thus the error per eigenvalue is bound by δ from above. Plugging into eq. (13) we obtain

$$\| |\Psi\rangle - |\tilde{\Psi}\rangle \|_2^2 \leq \underbrace{\sum_{i=1}^N |\beta_i|^2}_{=1} c^2 \kappa^2 \delta^2 = \delta^2 c^2 \kappa^2$$

and therefore

$$\| |\Psi\rangle - |\tilde{\Psi}\rangle \|_2 \leq c\kappa\delta.$$

An error of $\mathcal{O}(\varepsilon)$ is achieved with an QPE error $\mathcal{O}(\varepsilon/\kappa)$ and run time $\mathcal{O}(\kappa \text{ poly}(n)/\varepsilon)$.

For the measurement we want the ancilla qubit to be $|1\rangle$ for well conditioned A and $|well\rangle$ for arbitrary A . The probability of success is (using $\sum |\beta_i|^2 / |\lambda_i|^2 \simeq |A^{-1}b\rangle$)

$$p \geq \sum_{i: \lambda_i \geq \frac{1}{\kappa}} |\beta_i|^2 \left| \frac{1}{\lambda_i \kappa} \right|^2 = \mathcal{O}\left(\frac{1}{\kappa^2}\right).$$

Using *amplitude amplification* (similar to algorithm 5) we can increase the probability of success to $\mathcal{O}(1/\kappa)$. Hence, $\mathcal{O}(\kappa)$ is required for the procedure to get $|well\rangle$ with arbitrarily high probability.

5.8.4. Total computational complexity.

- state preparation: $\mathcal{O}(n)$, if QRAM is precomputed
- Hamiltonian simulation: If A is s -sparse, $\exp(iAt)$ can be implemented (up to an error ε in $\mathcal{O}(nst \text{ poly}(\log(st/\varepsilon)))$).
Note: this is the best result shown in literature. The simple method shown in the lecture has a complexity of $\mathcal{O}(ns^2t^2/\varepsilon)$.
- QPE: $\mathcal{O}(\text{poly}(n)\kappa/\varepsilon \cdot T_u)$
- Amplitude amplification: $\mathcal{O}(\kappa)$

In total, the complexity is $\mathcal{O}(\text{poly}(n)\kappa^2s/\varepsilon \cdot \text{poly}(\log(s/\varepsilon)))$. Compared to the CG method's $\mathcal{O}(\exp(n)s\sqrt{\kappa} \ln(2/\varepsilon))$, we see an exponential speed-up in n but a slowdown in κ, ε . Can we improve the complexity in κ, ε ?

5.9. Improvements on the HHL-algorithm. Comparing the CG method and the HHL algorithm we see that

CG: complexity vs. exactness $\mathcal{O}(\log(1/\varepsilon))$

HHL: Due to QPE only $\mathcal{O}(1/\varepsilon)$

5.9.1. *Goal.* Our goal is therefore to derive q-linear system algorithms which also have a logarithmic dependence on ε^{-1} but retain the exponential speedup in n .

5.9.2. *Idea.* Use a direct approximation of A^{-1} similar to Caley-Hamilton theorem. Let p be the characteristic polynomial of A with $p(A) = 0$

$$\begin{aligned} G\mathcal{P}_N &\implies A^N + \alpha_{N-1}A^{N-1} + \dots + \alpha_1A + \alpha_0\text{id} = 0, \quad \alpha_i \in \mathbb{C} \\ \iff A^{-1} &= -\frac{1}{\alpha_0} (A^{N-1} + \alpha_{N-1}A^{N-2} + \dots + \alpha_1\text{id}) = p_{N-1}(A). \end{aligned}$$

Determining α_i is too expensive but there might exist a polynomial $q_m \in \mathcal{P}_m$ with $m \ll N$ such that

$$A^{-1} \approx q_m(A)$$

and q_m can be implemented efficiently.

There are two possibilities to find q_m ,

- 1) using trigonometric polynomials with a Fourier approximation which is unitary and
- 2) using Chebyshev polynomials and minimizing $\|q_m(A)\|_2$.

We will now examine how the approximation of A^{-1} effects the final state.

Lemma 71. Let B be a Hermitian matrix with $\|B^{-1}\| \leq 1$ and D such that $\|B - D\| \leq \varepsilon < 1/2$. Then the states

$$|x\rangle := \frac{B|\psi\rangle}{\|B|\psi\rangle\|}, \quad |\tilde{x}\rangle := \frac{D|\psi\rangle}{\|D|\psi\rangle\|}$$

satisfy

$$\| |x\rangle - |\tilde{x}\rangle \| \leq 4\varepsilon.$$

Proof. Using the triangle inequality we obtain

$$\begin{aligned} \| |x\rangle - |\tilde{x}\rangle \| &= \left\| \frac{B|\psi\rangle}{\|B|\psi\rangle\|} - \frac{D|\psi\rangle}{\|D|\psi\rangle\|} \right\| \\ &\leq \left\| \frac{B|\psi\rangle}{\|B|\psi\rangle\|} - \frac{B|\psi\rangle}{\|D|\psi\rangle\|} \right\| + \frac{1}{\|D|\psi\rangle\|} \| \|B|\psi\rangle\| - \|D|\psi\rangle\| \| \\ &\leq \frac{\| \|D|\psi\rangle\| - \|B|\psi\rangle\| \|}{\|D|\psi\rangle\|} + \frac{1}{\|D|\psi\rangle\|} \| \|B|\psi\rangle\| - \|D|\psi\rangle\| \|. \end{aligned}$$

By assumption $1 \leq \|B|\psi\rangle\|$, hence using the triangle inequality once more we end up with

$$1 \leq \|B|\psi\rangle\| \leq \|D|\psi\rangle\| + \|(B - D)|\psi\rangle\| \leq \|D|\psi\rangle\| + \varepsilon$$

and therefore

$$\frac{\| \|D|\psi\rangle\| - \|B|\psi\rangle\| \|}{\|D|\psi\rangle\|} \leq \frac{\varepsilon}{\|D|\psi\rangle\|} + \frac{\varepsilon}{\|D|\psi\rangle\|} \leq 2 \frac{\varepsilon}{1 - \varepsilon} \leq 4\varepsilon.$$

□

We can apply theorem 71 to $B = A^{-1}$ and the approximation $D \approx A^{-1}$.

5.9.3. *Fourier approach.* We will now approximate A^{-1} as a linear combination of unitary operators, namely

$$A^{-1} \approx \sum_j \alpha_j e^{-iAt_j}.$$

We are therefore interested in

- how well can A^{-1} be approximated by a Fourier series and
- can it be implemented efficiently.

Without loss of generality, let $\alpha_j > 0$ for all j as the phase cannot be measured.

5.9.4. *Goal.* Our goal is the implementation of

$$M = \sum_j \alpha_j U_j$$

where U_j are unitary operators. Note that M itself does not have to be unitary.

Lemma 72. Let $M = \sum_j \alpha_j U_j$ with $\alpha_j > 0$ and U_j unitary. Furthermore, let V be the map defined by

$$V|0^m\rangle := \frac{1}{\sqrt{\alpha}} \sum_j \sqrt{\alpha_j} |j\rangle$$

with $\alpha = \sum_j \alpha_j$ and

$$U := \sum_j |j\rangle\langle j| \otimes U_j.$$

Then $W := V^H U V$ satisfies

$$W|0^m\rangle|\psi\rangle = \frac{1}{\alpha} M|\psi\rangle + |\Phi^\perp\rangle$$

for all states $|\psi\rangle$ with $\pi|\Phi^\perp\rangle := (|0^m\rangle\langle 0^m| \otimes \text{id})|\Phi^\perp\rangle = 0$.

Here the first term realizes M (later A^{-1}) and the second term is orthogonal to $|0^m\rangle$ inside the first register. Thus we can implement M by measuring $|0^m\rangle$ in the first register.

Proof.

$$\begin{aligned} W|0^m\rangle|\psi\rangle &= V^H U \left(\frac{1}{\sqrt{\alpha}} \sum_j \sqrt{\alpha_j} |j\rangle |\psi\rangle \right) \stackrel{\text{select } U}{=} V^H \left(\frac{1}{\sqrt{\alpha}} \sum_j \sqrt{\alpha_j} |j\rangle U_j |\psi\rangle \right) \\ &= \Pi V^H \left(\frac{1}{\sqrt{\alpha}} \sum_j \sqrt{\alpha_j} |j\rangle U_j |\psi\rangle \right) + (\text{id} - \Pi) V^H \left(\frac{1}{\sqrt{\alpha}} \sum_j \sqrt{\alpha_j} |j\rangle U_j |\psi\rangle \right) \\ &= (|0^m\rangle \otimes \text{id}) \left(\frac{1}{\sqrt{\alpha}} \sum_j \sqrt{\alpha_j} |j\rangle U_j |\psi\rangle \right) + |\Phi^\perp\rangle \end{aligned}$$

where we used the fact that $\Pi(\text{id} - \Pi) = 0$ for the last equality. Since Π projects the first register onto $|0^m\rangle$, and with the definition of V^H we get

$$W|0^m\rangle|\psi\rangle = \frac{1}{\alpha} |0^m\rangle \sum_j \alpha_j U_j |\psi\rangle + |\Phi^\perp\rangle.$$

□

Remark 73. • *Select U chooses U_i using a control register.*

- *The probability of success is $\|M|\psi\rangle\|^2/\alpha^2$. In our use case, $M = A^{-1}$, $|\psi\rangle = |b\rangle$ with QRAM, multiple $|b\rangle$ preparation is possible (rotation of $|b\rangle$), facilitating a quadratic speedup with high probability of $\mathcal{O}(\alpha/\|M|\psi\|)$ iterations.*
- *V is unitary and hence easy to implement.*
- *If the unitary operators U_i are easy to implement and the decomposition has $\mathcal{O}(\log N)$ terms, M can be implemented efficiently.*
- *The query-complexity of U is approximately the query complexity of the most expensive U_i . This is, in general, not desirable for the gate-complexity. In this special case however it is as all $U_j = \exp(-iA)^{t_j}$ have the same complexity.*
- *Using a diagonalization of $A = T^H D T$ we obtain*

$$T^H D^{-1} T = A^{-1} \stackrel{!}{\approx} \sum_j \alpha_j e^{-iA t_j} = T^H \sum_j \alpha_j e^{-iD t_j} T$$

and since $D = \text{diag}(\lambda_j)$ we only need an expansion of

$$f(x) = \frac{1}{x} \stackrel{!}{\approx} \sum_j \alpha_j e^{-i x t_j}$$

for $x \in \sigma(A)$ or $x \in \Omega \supseteq \sigma(A)$.

Similar to HHL, we only consider eigenvalues with good condition number ($\lambda_j \geq 1/\kappa$) and a scaling such that $\max_j \lambda_j = 1$. We look for an approximation of f on $D_k := [-1, 1] \setminus [-1/\kappa, 1/\kappa]$.

5.9.5. *Idea.* To approximate f we will

- 1) smooth f in a neighborhood of 0,
- 2) apply a Fourier expansion (integral),
- 3) and truncate the expansion by replacing the integral with a finite sum.

We end up with a function h of the form $\sum_j \alpha_j e^{-ixt_j}$ with

$$\sup_{x \in D_k} |f(x) - h(x)| \leq C\varepsilon.$$

?? implies that $h(A)|b\rangle$ (with normalization) is an approximation of $|x\rangle$.

Lemma 74. *The function*

$$h(x) := \frac{i}{\sqrt{2\pi}} \sum_{j=0}^{J-1} \sum_{k=-K}^K \Delta_y \Delta_z z_k e^{-\frac{z_k^2}{2}} e^{-ixy_j z_k}$$

where

$$y_j := j\Delta_y, \quad z_k := k\Delta_z, \quad J = \mathcal{O}\left(\frac{\kappa}{\varepsilon} \log \frac{\kappa}{\varepsilon}\right), \quad K = \mathcal{O}\left(\kappa \log \frac{\kappa}{\varepsilon}\right),$$

$$\Delta_y = \mathcal{O}\left(\frac{\varepsilon}{\sqrt{\log(\kappa/\varepsilon)}}\right), \quad \text{and} \quad \Delta_z = \mathcal{O}\left(\kappa \frac{1}{\sqrt{\log(\kappa/\varepsilon)}}\right)$$

satisfies

$$\sup_{x \in D_k} \left| h(x) - \frac{1}{x} \right| \leq C\varepsilon.$$

Proof. Defining $g(y) := y \exp(-y^2/2)$, and using substitution we find

$$\frac{1}{x} = \int_0^\infty g(xy) dy$$

since

$$\int_0^\infty g(x) dy = 1.$$

We choose g in this way as it decays fast, is smooth, and because

$$\mathcal{F}(g) = -ig$$

g is an eigenfunction of the Fourier expansion with eigenvalue $-i$. This implies

$$g = -i\mathcal{F}(g) = \frac{i}{\sqrt{2\pi}} \int_{\mathbb{R}} z e^{-\frac{z^2}{2}} e^{-iyz} dz \tag{15}$$

$$\implies \frac{1}{x} = \int_0^\infty \int_{\mathbb{R}} z e^{-\frac{z^2}{2}} e^{-ixyz} dz dy. \tag{16}$$

Note that $h(x)$ is a Riemann sum for the truncated integral in eq. (15).

1. Step Summation/integration in y : Using the geometric series it follows that

$$h(x) = \frac{i\Delta_y}{\sqrt{2\pi}} \sum_{k=-K}^K \Delta_z z_k e^{-\frac{z_k^2}{2}} \frac{1 - e^{-ixy_j z_k}}{1 - e^{-ix\Delta_y z_k}} \tag{17}$$

and

$$\frac{1}{x} = \frac{1}{\sqrt{2\pi}} \cdot \frac{1}{x} \int_{\mathbb{R}} e^{-\frac{z^2}{2}} dz.$$

2. Step Approximation of $1/x \int_{\mathbb{R}} e^{-\frac{z^2}{2}} dz$ with an infinite Riemann-sum: Using Poisson's summation formula

$$\sum_{k=-\infty}^{\infty} f(k) = \sqrt{2\pi} \sum_{k=-\infty}^{\infty} \hat{f}(2\pi k)$$

we obtain

$$\frac{1}{\sqrt{2\pi}} \sum_{k=-\infty}^{\infty} \Delta_z e^{-\frac{z_k^2}{2}} = \sum_{k=-\infty}^{\infty} e^{-\left(\frac{2\pi k}{\Delta_z}\right)^2/2} = 1 + \sum_{|k| \geq 1} e^{-\left(\frac{2\pi k}{\Delta_z}\right)^2/2} \quad (18)$$

with mesh size Δ_z and therefore

$$\left| \frac{1}{x} \left(1 - \frac{1}{\sqrt{2\pi}} \sum_{k=-\infty}^{\infty} \Delta_z e^{-\frac{z_k^2}{2}} \right) \right| \stackrel{(18)}{\leq} \frac{1}{|x|} \cdot 2 \sum_{k=1}^{\infty} e^{-\frac{2\pi^2 k^2}{\Delta_z^2}} = \frac{1}{|x|} \frac{2}{e^{2\pi^2 \Delta_z^2} - 1} \leq CK \cdot \varepsilon e^{-\kappa^2 \log \kappa}.$$

After choosing Δ_z and with $1/|x| \leq \kappa$ we have

$$CK \cdot \varepsilon e^{-\kappa^2 \log \kappa} \leq \tilde{C} \varepsilon.$$

3. Step Truncating the series expansion: By adding $(1 - \exp(-ixy_j z_k))$, using the triangle inequality, Poisson summation formula, and estimating the rest of the series by choosing K we obtain

$$\left| \frac{1}{\sqrt{2\pi} x} \left(\sum_{k=-\infty}^{\infty} \Delta_z e^{-\frac{z_k^2}{2}} - \sum_{k=-K}^K \Delta_z e^{-\frac{z_k^2}{2}} (1 - e^{-ixy_j z_k}) \right) \right| \leq C\varepsilon. \quad (19)$$

Deriving eq. (19) is a lengthy calculation and is therefore omitted at this point. It can, however, be found in the literature.

4. Step Since

$$\left| \frac{1}{1 - e^{-ix}} - \frac{1}{x} \right| < 1 \quad \forall x \in [-1, 1]$$

we have

$$\frac{i\Delta_y z_k}{1 - e^{-ix\Delta_y z_k}} - \frac{1}{x} \leq \Delta_y |z_k|$$

and hence

$$\begin{aligned} \left| h(x) - \frac{1}{\sqrt{2\pi} x} \sum_{k=-K}^K \Delta_z e^{-\frac{z_k^2}{2}} (1 - e^{-ixy_j z_k}) \right| &\stackrel{(17)}{\leq} \frac{2}{\sqrt{2\pi}} \left| \sum_{k=-K}^K \left(\frac{i\Delta_y z_k}{1 - e^{-ix\Delta_y z_k}} - \frac{1}{x} \right) \cdot \Delta_z e^{-\frac{z_k^2}{2}} \right| \\ &\leq \sqrt{\frac{2}{\pi}} \Delta_y \underbrace{\sum_{k=-K}^K \Delta_z e^{-\frac{z_k^2}{2}}}_{\leq \int_0^\infty \exp(-z^2/4) dz} = C\Delta_y \Delta_z \leq C\varepsilon \end{aligned}$$

up to log-terms.

5. Step By combining the steps 2-4 with the triangle inequality leads to

$$\left| h(x) - \frac{1}{x} \right| \leq C\varepsilon.$$

□

5.9.6. Complexity.

Theorem 75. *The quantum linear system of equations can be solved for $\exp(-iAt)$ with $t = \mathcal{O}(\kappa \log(\kappa/\varepsilon))$ and accuracy $\mathcal{O}(\varepsilon/(\kappa \sqrt{\log(\kappa/\varepsilon)}))$, by applying Hamiltonian simulation $\mathcal{O}(\kappa \sqrt{\log(\kappa/\varepsilon)})$ times. The gate-complexity is given by*

$$\mathcal{O}\left(s\kappa^2 \log^{2.5}\left(\frac{\kappa}{\varepsilon}\left(\log N + \log^{2.5}\left(\frac{\kappa}{\varepsilon}\right)\right)\right)\right).$$

Proof. At this point, we will only give a sketch of the proof. The full proof can be found in the literature.

We have to implement U and V . For sparse A we can use Hamiltonian simulation with

$$\mathcal{O}\left(st\left(\log N + \log^{2.5}\left(\frac{t}{\varepsilon}\right)\right)\log\frac{t}{\varepsilon}\right)$$

for U and select U_i . The choice of t implies a certain accuracy. Lastly, Hamiltonian simulation has to be applied $\mathcal{O}(\kappa \sqrt{\log(\kappa/\varepsilon)})$ times. \square

5.9.7. Chebychev approach.

5.9.8. *Idea.* In this section we will replace the Fourier expansion by Chebychev polynomials.

5.9.9. *Problem.* The chebychev polynomials are not unitary. The solution is *block-encoding* ($n + 1$ -qubit block encoding) as mentioned in the Chapter ??.

Remark 76. \bullet *LCU (without amplitude amplification) is a special case of block-encoding.*

- \bullet *Block encoding for an arbitrary A is too expensive. However, for a sparse A it is efficiently feasible.*

5.9.10. *Goal.* Our goal is to implement block-encoding for A^{-1} .

Theorem 77. *Let $p \in \mathcal{P}_d$ with $\|p\|_{\infty,[-1,1]} \leq 1/4$ be a polynomial of degree $\leq d$ and U block-encoding of A with $n + a$ -qubits.*

A block-encoding of $P(A)$ with $n + \mathcal{O}(a)$ qubits can be achieved with d applications of U, U^{-1} , one application of controlled- U and $\mathcal{O}(ad)$ elementary gates (2-qubits).

Example 78. *Application of theorem 77 To approximate $f(x) = 1/x$ by $p(x)$ we use the fact that for $\tilde{d} = \mathcal{O}(\kappa^2 \log(\kappa/\varepsilon))$ we have*

$$\sup_{x \in D_k} \left| \underbrace{\frac{1 - (1 - x^2)^{\tilde{d}}}{x}}_{\in \mathcal{P}_{2\tilde{d}-1}} - \frac{1}{x} \right| \leq \frac{\varepsilon}{2}.$$

Now using the Chebychev polynomials T_j we get

$$\frac{1 - (1 - x^2)^{\tilde{d}}}{x} = \sum_{j=0}^{2\tilde{d}-1} \alpha_j T_j.$$

Furthermore, truncating the expansion at $d = \mathcal{O}(\kappa \log(\kappa/\varepsilon))$ results in an error $\leq \varepsilon/2$. Theorem 77 provides an efficient implementation of $P_d(A)$. Through the use of block-encoding we obtain the state $|\tilde{x}\rangle$ with $|\|\tilde{x}\rangle - |x\rangle| \leq C\varepsilon$.

5.9.11. *Complexity.* In total, the gate-complexity is given by

$$\mathcal{O}\left(s\kappa^2 \log^2\left(\frac{s\kappa}{\varepsilon}\right) \left(\log N + \log^{2.5}\left(\frac{s\kappa}{\varepsilon}\right)\right)\right),$$

which is better than the Fourier approach. However, the Chebychev approach needs direct sparse access which is not possible for arbitrary matrices. Using Hamiltonian simulation instead of the Fourier approach is therefore more general.

5.10. **Newton's method on quantum computers.** The goal in this section is to find zeros of a (nonlinear) function $f : \mathbb{C}^n \rightarrow \mathbb{C}^n$, i.e., find $x^* \in \mathbb{C}^n$ such that

$$f(x^*) = 0.$$

The main issue here is that the function f can be non-linear and thus, in general, is not unitary. In the following, we restrict f to be polynomial, as polynomials actually can be realized on quantum computers by means of tensorization of the same state.

Consider a quantum state $|v\rangle$, then the basis expansion of the tensor product reads as

$$|v\rangle|v\rangle = \sum_{j=0}^{2^n-1} \sum_{k=0}^{2^n-1} v_j v_k |j\rangle|k\rangle = \sum_{j=0}^{2^n-1} v_j^2 |j\rangle|j\rangle + \sum_{j=0}^{2^n-1} \sum_{k=0, k \neq j}^{2^n-1} v_j v_k |j\rangle|k\rangle.$$

Now, the amplitudes in the first sum on the right-hand side encode the information of squaring, i.e., they offer a method of realization for the function $f(x) = x^2$. Applying a tensorized CNOT (or in other words a block encoding), we can extract this information by partial measurement

$$\text{CNOT}^{\otimes n} |v\rangle|v\rangle = \sum_{j=0}^{2^n-1} \sum_{k=0}^{2^n-1} v_j v_k |j\rangle|j \oplus k\rangle = |v^2\rangle|0^n\rangle + |\perp\rangle,$$

where the last summand is orthogonal to $\mathbb{C}^{2^n} \otimes |0^n\rangle$. Thus, measurement of the last n -qubit register and only keeping the result if we measure $|0^n\rangle$, gives a realization of squares.

Theorem 79. *Let f be a multivariate polynomial. Assume that Newton's method converges quadratically for initial guess $x_0 \in \mathbb{C}^n$ on a classical computer. Assume that f and $|x_0\rangle$ can be realized by quantum circuits. Then, there exists a quantum algorithm that approximates x^* up to tolerance $\varepsilon > 0$ and failure probability $\delta > 0$ with runtime*

$$\mathcal{O}(\varepsilon^{-(1+\mu)} \log \delta^{-1})$$

for any $\mu > 0$.

5.11. **Solving Linear Differential Equations.** Every linear differential equation can be reduced to a first order system

$$x'(t) = Ax(t) + b(t) \quad \in \mathbb{R}^{N_x} x(0) = x_0$$

where $A \in \mathbb{R}^{N_x \times N_x}$ is sparse, $b \in \mathbb{R}^{N_x}$, $x_0 \in \mathbb{R}^{N_x}$ and $s \in \mathbb{N}$.

Idea 1: Lie-Trotter approach:

- exponential cost in t
- no inhomogeneous equations

Idea 2: Space-time approach [1] Feynman's clock: Encode time in basis states $|j\rangle$ and produce output state

$$|\psi(t)\rangle = \sum_{j=0}^{N_t} |j\rangle |x_j\rangle$$

where $x_j \approx x(t_j)$ and

$$\begin{aligned} N_t &= \frac{T}{\tau} \dots && \text{number of time steps} \\ T \dots &&& \text{final time} \\ \tau \dots &&& \text{time step size.} \end{aligned}$$

Since the probability of measuring $x(T) \approx x_{N_t}$ is small, we can extend the ODE beyond T with

$$A(t) := \text{id}, \quad b(t) := 0 \quad \forall (T, 2T].$$

Thus, increasing the probability as $x(t) = x(T)$ for $t \in [T, 2T]$.

Example 80. Forward Euler A simple example is the forward Euler method given by

$$\frac{x_{j+1} - x_j}{\tau} = A(t_j)x_j + b(t_j) \quad \forall t \in [0, 2T].$$

We define the vectors \bar{x} , \bar{b} and the matrix $\underline{A} \in \mathbb{R}^{2N_t N_x \times 2N_t N_x}$ as

$$\bar{x} := \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{2N_t} \end{pmatrix}, \quad \bar{b} := \begin{pmatrix} x_0 \\ b_0\tau \\ b_1\tau \\ \vdots \\ b_{N_t}\tau \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad \underline{A} := \begin{pmatrix} \text{id} & 0 & \dots & 0 \\ -(\text{id} + A\tau) & \text{id} & & \\ & \ddots & & \\ & & -(\text{id} + A\tau) & \text{id} \\ & & & -\text{id} & \text{id} \\ & & & & \ddots & 0 \\ & & & & & -\text{id} & \text{id} \end{pmatrix}$$

leads to the formulation $\underline{A}\bar{x} = \bar{b}$.

The local Euler error $\approx \tau^2$ and hence the error at end-time is $2N_t\tau^2 \approx T^2/N_t$. To achieve error ε , we need $N_t \approx T^2/\varepsilon$. Using the HHL-algorithm requires a bounded condition number. Consider

$$\begin{pmatrix} 1 & & & 0 \\ -1 & \ddots & & \\ & \ddots & \ddots & \\ 0 & & -1 & 1 \end{pmatrix} \underbrace{\begin{pmatrix} 1 \\ 2 \\ \vdots \\ n \end{pmatrix}}_x = \underbrace{\begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}}_b$$

with

$$\|\underline{A}^{-1}\| \geq \frac{\|x\|}{\|b\|} = \frac{\sqrt{\sum_{i=1}^n i^2}}{\sqrt{\sum_{i=1}^n 1}} \approx \frac{n^{\frac{3}{2}}}{n^{\frac{1}{2}}} = n.$$

Hence the condition number κ of \underline{A} is $\kappa \approx 2N_t N_x \approx T^2$ and thus the HHL algorithm requires at least $\kappa^2 \approx T^4$ operations.

5.12. Multi-step methods. The idea is to use the information of multiple time steps to increase the accuracy. The multistep method is given by

$$\sum_{l=0}^k d_l x_{j+l} = \tau \sum_{l=0}^k \beta_l (A(t_{j+l})x_{j+l} + b(t_{j+l}))$$

and its stability by

$$\rho(\xi) = \sum_{j=0}^k \alpha_j \xi^j, \quad \sigma(\xi) = \sum_{j=0}^k \beta_j \xi^j.$$

Let $R_j(\mu)$ denote the roots of $\rho(\xi) - \mu\sigma(\xi) = 0$ and define

$$S := \left\{ \mu \in \mathbb{C} : \begin{array}{l} \text{all roots of } R_j(\mu) \text{ satisfy } |R_j(\mu)| \leq 1 \\ \text{multiple roots } R_j(\mu) \text{ satisfy } |R_j(\mu)| < 1 \end{array} \right\}.$$

If all roots of σ are elements of S , the method is stable at infinity. In matrix form the method reads

$$\underline{A} = \begin{cases} \underline{A}_{j,j} = & 0 \leq j \leq k, \quad N_t < j \leq 2N_t \\ \underline{A}_{j,j-1} = -(\text{id} + A\tau) & 1 \leq j < k \\ \underline{A}_{j,j-k+l} = \alpha_l \text{id} - \beta_l A\tau & k \leq j \leq N_t, \quad 0 \leq l \leq k \\ \underline{A}_{j,j-1} = -\text{id} & N_t < j \leq 2N_t \end{cases}$$

$$\bar{b} = \begin{cases} \bar{b}_0 = x_0 \\ \bar{b}_j = bh & 1 \leq j < k \\ \bar{b}_j = \sum_{l=0}^k \beta_l bh & k \leq j \leq N_t \\ \bar{b}_j = 0 & N_t < j \leq 2N_t. \end{cases}$$

Assume there are oracles O_A and $O_{\mathcal{F}}$ such that $O_A|j, l\rangle|z\rangle = |j, l\rangle z \oplus A_{j,l}$ and $O_{\mathcal{F}}|j, l\rangle = |j, f(j, l)\rangle$ where $A_{j, l}$ is in binary representation and $f(j, l)$ is the l -th non-zero in column j . A similar oracle for the l -th non-zero in row j is also needed.

Lemma 81. *There holds $\|\underline{A}\| \lesssim 1$ if $\tau \lesssim 1/\|A\|$.*

Proof. We write \underline{A} as sum of block diagonal matrices

$$\underline{A} = \sum_{k=0}^l \underline{A}_k,$$

with norm

$$\begin{aligned} \|\underline{A}_0\| &\leq \max\{1, |\alpha_k| + |\beta_k|h\|A\|\}, \\ \|\underline{A}_1\| &\leq \max\{1 + h\|A\|, |\alpha_{k-1}| + |\beta_{k-1}|h\|A\|\}, \\ \|\underline{A}_l\| &\leq |\alpha_l| + |\beta_l|h\|A\|, \quad \forall 2 \leq l \leq k. \end{aligned}$$

Thus the norm of \underline{A} is bounded by

$$\|\underline{A}\| \leq \sum_{l=0}^k \|\underline{A}_k\| \lesssim k \lesssim 1.$$

□

Lemma 82. *Assume that $A = VDV^{-1}$ with eigenvalues λ_i such that $|\arg(-\lambda_i)| \leq \alpha$. Furthermore, assume the multi-step method is $A(\alpha)$ -stable ($S \supseteq \{\lambda \in \mathbb{C} : |\arg(-\lambda)| <$*

$\alpha, \lambda \neq 0\}$).

Then,

$$\|\underline{A}^{-1}\| \lesssim N_t \kappa_V,$$

where $\kappa_V := \|V\| \|V^{-1}\|$ is the condition number of V .

Proof. Let \underline{V} denote the block diagonal matrix

$$\underline{V} = \begin{pmatrix} V & & & \\ & V & & \\ & & \ddots & \\ & & & V \end{pmatrix}$$

and \underline{D} the matrix \underline{A} where we replace A by D . Then, $\underline{A} = \underline{V} \underline{D} \underline{V}^{-1}$ and $\|\underline{A}^{-1}\| \leq \kappa_V \|\underline{D}^{-1}\|$.

It remains to estimate $\|\underline{D}^{-1}\|$. We write \underline{D} as

$$\underline{D} = \sum_{l=0}^k \underline{D}_l$$

with (off-diagonal) block-matrices \underline{D}_l . Depending on l we have

$l = 1$:

$$\underline{D}^{-1} = (\underline{D}_0 + \underline{D}_1)^{-1} = \underline{D}_0^{-1} (\text{id} + \underline{D}_1 \underline{D}_0^{-1})^{-1} = \underline{D}_0^{-1} \sum_{k=0}^{\infty} (-\underline{D}_1 \underline{D}_0^{-1})^k$$

Note that $\underline{D}_1 \underline{D}_0^{-1}$ is of the form

$$\underline{D}_1 \underline{D}_0^{-1} = \begin{pmatrix} 0 & & & \\ \star & \ddots & & \\ & \ddots & \ddots & \\ & & & \star & 0 \end{pmatrix}$$

and therefore $\|\underline{D}_1 \underline{D}_0^{-1}\| = 0$ for $k \geq 2N_t$. Furthermore, the off-diagonal entries have the form

$$(1 + C\tau)^k \lesssim 1, \quad \forall k \leq \frac{1}{\tau} = N_t$$

and hence, $\|\underline{D}^{-1}\| \lesssim N_t$.

$l > 1$: (sketch)

$\underline{D}y = r$ corresponds to the discretization of the system

$$y^{(j)'}(t) = \lambda_j y^{(j)}(t) + r^{(j)}(t).$$

Since the method is α -stable, the numerical approximation $y_i^{(j)}$ cannot grow unless forced by r^j . Let

$$\left(y_i^{(j,l)} \right)_{i=1}^{N_t}$$

denote the solution with right hand side

$$\left(r_i^{(j)} \delta_{ik} \right)_{i=1}^{N_t}$$

and initial condition $x_0\delta_{k0}$, i.e.,

$$y_i^{(j)} = \sum_{k=0}^{N_t} y_i^{(j,k)}.$$

Stability shows $|y_i^{(j,k)}| \lesssim |r_k^{(j)}|$ for $i \geq k$ and hence

$$\|y^{(j)}\| \lesssim \sum_{k=0}^{N_t} \sqrt{N_t - k} |r_k^{(j)}| \lesssim N_t \underbrace{\sqrt{\sum_{k=0}^{N_t} |r_k^{(j)}|^2}}_{\|r^{(j)}\|}$$

and

$$\|y\| \lesssim N_t \|r\|.$$

□

Theorem 83. *Under the above assumptions, the HHL-algorithm produces a state proportional to*

$$|\psi\rangle = \sum_{j=0}^{N_t} |j\rangle |x_j\rangle$$

with error ε in

$$\mathcal{O}\left(\log N_x s^{\frac{q}{2}} (\|A\|T)^{2+\frac{2}{p}} \kappa_V^5 \left(\|x_0\| + \frac{\|b\|}{\|A\|}\right) \varepsilon^{-2}\right)$$

calls to the oracles for A, b , and x_0 .

Theorem 84. [2] *Suppose $A = VD V^{-1}$ with $D \leq 0$. Assume $A(t) = A$ and $b(t) = b$. There exists a quantum algorithm that produces*

$$\frac{x(T)}{\|x(T)\|}$$

up to an error ε with

$$\mathcal{O}\left(\kappa_V s \rho T \|A\| \text{poly}\left(\log\left(\kappa_V s \rho \beta T \frac{\|A\|}{\varepsilon}\right)\right)\right)$$

query calls, where

$$\rho := \max_{t \in [0, T]} \frac{\|x(t)\|}{\|x(T)\|}$$

$$\beta := \frac{\|x_0\| + T\|b\|}{\|x(T)\|}.$$

REFERENCES

- [1] Dominic W Berry. High-order quantum algorithm for solving linear differential equations. *Journal of Physics A: Mathematical and Theoretical*, 47(10):105301, February 2014.
- [2] Dominic W. Berry, Andrew M. Childs, Aaron Ostrander, and Guoming Wang. Quantum algorithm for linear differential equations with exponentially improved dependence on precision. *Communications in Mathematical Physics*, 356(3):1057–1081, October 2017.
- [3] R. de Wolf. Quantum computing: Lecture notes. *arXiv:1907.09415v5*, 2019.
- [4] M. Kaltenböck. Kernreproduzierende hilberträume. *Vorlesungsskript, TU Wien*, 2024.
- [5] L. Lin. Lecture notes on quantum algorithms for scientific computation. *arXiv:2201.08309*, 2022.

- [6] Gerald Teschl. *Mathematical methods in quantum mechanics*, volume 99 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2009. With applications to Schrödinger operators.