# Quantum Computing - Summary

**WS2023/24**

Manuel Waibel

February 15, 2024

## Contents

# 0 Introduction

## 0.1 Qubit

Is a "bit" that can be $|0\rangle$, $|1\rangle$or in a mixed state (superposition). If measured it can be between $|0\rangle$and $|1\rangle$.

$$\alpha|0\rangle + \beta|1\rangle$$

$$|\alpha|^2 + |\beta|^2 = 1$$

$n$ such qubits can be in a superposition of $2^n$ states: $\alpha_0|00\rangle + \alpha_1|01\rangle + \alpha_2|10\rangle + \alpha_3|11\rangle$. A quantum computer can deal with these $2^n$ states in parallel.

A qubit can also be written in coordinate form, since the element $|0\rangle$and $|1\rangle$of the basis are fixed:

- for $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$ we write then $|\Psi\rangle = \begin{pmatrix}\alpha\\ \beta\end{pmatrix}$
- Hence: $|0\rangle = \begin{pmatrix}1\\ 0\end{pmatrix}$ and $|1\rangle = \begin{pmatrix}0\\ 1\end{pmatrix}$

## 0.2 Entaglement

Explenation video: "Quantum Entanglement & Spooky Action at a Distance"

A system is entangled if the corresponding quantum state cannot be written as a tensor product of its components.
For example in a 2 qubit system we either measure $|00\rangle$ or $|11\rangle$: $\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$
This is called a **Bell state** or ERP pair.

## 0.3 Problems with current quantum computers

Problems when circuits run on a quantum computer: **Noise**

Two sources of noise:

1. **Gate infidelity:** User-specified gate does not precisely correspond to the physically implemented gate
2. **Decoherence:** Gradually over time, a quantum computer looses its "quantumness", e.g., due to interaction with the environment. Then it behaves more like a classical computer.

Both effects limit the depth of quantum circuits in practice.

# 1 Principles of Quantum Mechanics $I$

## 1.1 Classes of Operators

$T$ is a linear operator with $T : V \mapsto V$.
$T^* : V \mapsto V$ is defined as: $\langle x, Ty\rangle = \langle T^*x, y\rangle \qquad \forall x, y \in V$
For unitary operators it holds that $T^* = T^{-1}$ and further more $\langle x, y\rangle = \langle Ix, y\rangle = \langle T^*Tx, y\rangle = \langle Tx, Ty\rangle$ (unitary operators preserve the inner product).

$T$ is ...

- **self adjoined/hermitian**, if $T = T^*$ $(Tx = T^*x, \forall x \in V)$
  $\rightarrow$ Eigenvalues are real numbers
- **unitary**, if $TT^* = T^*T = I$ $(TT^*x = T^*Tx = x, \forall x \in V)$
- **(orthogonal) projection**, if $T$ is self adjoined and $T = T^2$

**Linear operator:** A linear operator preserves vector addition and scalar multiplication

$$T(\lambda x + \mu y) = \lambda T(x) + \mu T(y)$$

**Unitary operator:** A linear operator that preserves the norm of a quantum state and is also reversible.
Mathematically an operator $U$ statisfies following property, if it is unitary:

$$U^\dagger U = UU^\dagger = I$$

Where $U^\dagger$ is the adjoint of $U$, meaning the conjugate transpose.

$T^*$ is given in terms of the matrix $A^* = (\bar{A})^T = (\bar{a_{ji}})$, e.g.:

$$A = \begin{pmatrix} 1+3i & 2i \\ 1+i & 1-4i \end{pmatrix}$$

$$A^* = \begin{pmatrix} 1-3i & 1-i \\ -2i & 1+4i \end{pmatrix}$$

## 1.2 Bra- & Ket-Notation

The basic idea of this notation is that the inner product $\langle x, y\rangle$ of two vectors $x$, $y$ can be seen as the application of the "bra-vector" $\langle x|$ to the "ket-vector" $|y\rangle$, i.e. one writes $\langle x|y\rangle$ for $\langle x, y\rangle$.

Ket-vectors correspond to the usual vectors, whilst bra-vectors are seen as elements of a "dual space".

$$\langle\Psi| = \begin{pmatrix} x_1 & x_2 & \cdots & x_n \end{pmatrix}$$

$$|\Psi\rangle = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

# 2 Principles of Quantum Mechanics II & Quantum Gates

## 2.1 Postulates of Quantum Mechanics

### 2.1.1 Postulate I - Description of the state of a system

The state of an isolated physical system is completely described in terms of a state vector in a complex Hilbert space (state space). [1]

### 2.1.2 Postulate II - Time evolution of a system

The time evolution of a closed system is described by a unitary transformation on the initial state. [2]

$$|\psi(s+t)\rangle = U(t)|\psi(s)\rangle$$

Where $U(t)$ is a unitary operator, $|\psi(s)\rangle$ is the state of the system at time $s$ and $|\psi(s+t)\rangle$ is the state of the system at time $s+t$.

The time evolution can alternatively also be described as follows:

The time evolution of the state vector $|\psi(t)\rangle$ is governed by the **Schrödinger equation**, where $H(t)$ is a Hamiltonian operator[3] (the observable associated with the total energy of the system).

$$i\hbar\frac{\delta}{\delta t}|\psi(t)\rangle = H(t)|\psi(t)\rangle$$

where:

[1] Postulate 1 Wikipedia
[2] Postulate 3 Wikipedia
[3] Hamiltonian = total energy of a system, including both kinetic energy and potential energy

| | |
|---|---|
| $i$ | Imaginary number |
| $\hbar$ | Reduced planck's constant $\hbar = \frac{h}{2\pi}$ |
| $\frac{\delta}{\delta t}$ | Partial derivation |
| $H(t)$ | Hamiltonian operator (sometimes also $\hat{H}(t)$) |
| $\|\psi(t)\rangle$ | State vector |

### 2.1.3 Postulate III - Measurement on a system

Measurements are described by self-adjoint operators, called observables. Each observable $M$ has a spectral representation.

Possible values of measurements are given by the eigenvalues of $M$.
If directly before the measurement the system is in state $|\Psi\rangle$, then $p(m) = \langle\Psi|P_m|\Psi\rangle$, where $P_m$ is the projection, gives the probability to measure the value $m$ (square the amplitude).
After the measurement of $m$, the system is in state $\frac{1}{\sqrt{p(m)}}P_m|\Psi\rangle$, where the function $p(\cdot)$ satisfies the boundary condition $\sum_m p(m) = \sum_m \langle\psi|P_m|\Psi\rangle = 1$ (the sum of all probablities of all states must be 1).

Two observables also obey the Heisenberg uncertainty relation, which states, that the more precise we measure the phase/position of a qubit, the less precise (or more uncertain) its amplitude/momentum get.

### 2.1.4 Postulate IV - Superposition Postulate

The state space of a composite system $S$ is given by the tensor product of its parts.
That is, if $S$ consists of $n$ subsystems $S_1, ..., S_n$ and each $S_i$ is in state $|\Psi\rangle_i (i = 1, ..., n)$, then the state vector $|\Psi\rangle$of the overall system $S$ is given by

$$|\Psi\rangle_1 \otimes \cdots \otimes |\Psi\rangle_n.$$

## 2.2 Quantum Registers

Quantum registers are state vectors, that can be represented, by taking the tensor product ($\otimes$) of some (state) vectors.

$$|00\rangle = |\psi_0^1\rangle \otimes |\psi_0^2\rangle = \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}1\\0\\0\\0\end{pmatrix}$$

$$|01\rangle = |\psi_0^1\rangle \otimes |\psi_1^2\rangle = \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\1\\0\\0\end{pmatrix}$$

$$|10\rangle = |\psi_1^1\rangle \otimes |\psi_0^2\rangle = \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\0\\1\\0\end{pmatrix}$$

$$|11\rangle = |\psi_1^1\rangle \otimes |\psi_1^2\rangle = \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\0\\0\\1\end{pmatrix}$$

### 2.2.1 Entanglement

Entaglement of qubits/qubit-registers is used to execute "calculations" simultaniously. By measuring the property of one state, we immediately know the state of the other.

A system is entangled if the corresponding quantum state cannot be written as a tensor product of its components (Postulate IV).
For example in a 2 qubit system we either measure $|00\rangle$ or $|11\rangle$: $\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$

## 2.3 Quantum Gates

Unitary operators over $n$-qubit quantum registers are called $n$-qubit quantum gates. Quantum gates are represented by unitary $2^n \times 2^n$-matrices.

### 2.3.1 NOT gate (1-bit gate)

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

**Example:**

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

Application of the $\sqrt{NOT}$ gate results in a quantum state that neither corresponds to the classical bit 0 nor the classical bit 1:

$$\begin{aligned} \sqrt{NOT}|0\rangle &= \sqrt{NOT} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ &= \frac{1}{2} \begin{pmatrix} 1+i \\ 1-i \end{pmatrix} \\ &= \frac{1+i}{2} \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \frac{1-i}{2} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ &= \frac{1+i}{2}|0\rangle + \frac{1-i}{2}|1\rangle \end{aligned}$$

Applying the $\sqrt{NOT}$ gate multiple times reveals the classical $NOT$ gate:
$\sqrt{NOT} \cdot \sqrt{NOT} = NOT$.

The $X/NOT$ gate is a Pauli matrix. The other matrices include:

$$Y := \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

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

All Pauli gates applied to themselves reverse each other: $X^2 = Y^2 = Z^2 = I$ (where $I$ is the identity matrix).

### 2.3.2 Hadamard gate (1-bit gate)

The Hadamard gate is one of the most useful gates in quantum computing.

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

Like the $\sqrt{NOT}$ gate, it maps a computational basis into a superposition of states:

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

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

With the application of the Hadamard gate to $n$ qubits, we can generate $2^n$ equaly distributed superpositions of $|0\rangle$and $|1\rangle$(every possible combination of bit strings of $n$ bits).

The states $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ often also are called the "plus" and "minus" state:

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

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

### 2.3.3 Phase, $T$- and Rotation gate (1-bit gate)

$$R_X(\alpha) = e^{-i\alpha X/2} = \begin{pmatrix} \cos(\alpha/2) & -i\sin(\alpha/2) \\ -i\sin(\alpha/2) & \cos(\alpha/2) \end{pmatrix}$$
$$R_Y(\alpha) = e^{-i\alpha Y/2} = \begin{pmatrix} \cos(\alpha/2) & -\sin(\alpha/2) \\ \sin(\alpha/2) & \cos(\alpha/2) \end{pmatrix}$$
$$R_Z(\alpha) = e^{-i\alpha Z/2} = \begin{pmatrix} e^{-i\alpha/2} & 0 \\ 0 & e^{i\alpha/2} \end{pmatrix}$$
$$Ph(\delta) = e^{i\delta} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \text{ "global phase shift"}$$

### 2.3.4 $CNOT$ gate (2-bit gate)

The $CNOT$-gate is a controlled $NOT$-gate.

$$CNOT := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

It works as follows:

$$CNOT|00\rangle = |00\rangle$$
$$CNOT|01\rangle = |01\rangle$$
$$CNOT|10\rangle = |11\rangle$$
$$CNOT|11\rangle = |10\rangle$$

The first bit is the control bit: if set, then the second qubit is inverted.

The $CNOT$ gate is self-adjoint.

Special $CNOT$ gates: $CNOT_{12} \rightarrow$ control qubit 1 and target qubit 2.

The Toffoli gate is a special case of the $CNOT$ gate. It is esentially a $CCNOT$ gate (so two control bits): $|110\rangle \mapsto |111\rangle$ and $|111\rangle \mapsto |110\rangle$ and $|xyz\rangle \mapsto |xyz\rangle$ otherwise.

### 2.3.5 $SWAP$ gate (2-bit gate)

The $SWAP$-gate (as the name suggests) swaps two bits: $SWAP|xy\rangle \mapsto |yx\rangle$

$$SWAP := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The $SWAP$ gate can be derived from different versions from the $CNOT$ gate:
$CNOT_{12}CNOT_{21}CNOT_{12}$

The $SWAP$ gate is self-adjoint.

Special case: Fredkin or $CSWAP$ gate:
$|0yz\rangle \mapsto |0yz\rangle$ and $|1yz\rangle \mapsto |1zy\rangle$

$$CSWAP := \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

## 2.4 Universality

The Pauli matrices $Y$ and $Z$ are universal, meaning, that each 1-qubit gate $U$ can be constructed out of a combination of $X$, $Y$ and $Z$ gates:

$$U = e^{i\alpha} R_Z(\beta) R_Y(\gamma) R_Z(\delta)$$

for suitable $\alpha, \beta, \gamma, \delta \in \mathbb{R}$.

Each $n$-qubit gate $U$ ($n > 1$) can be represented in terms of $R_X(\cdot), R_Y(\cdot), R_Z(\cdot), Ph(\cdot)$ and $CNOT$.

# 3 Preparatory concepts of the quantum algorithms

## 3.1 Gates with multiple control qubits

We can/could construct a $CNOT$ gate with $n$ control qubits by using $n-1$ ancillary (temporary) qubits to connect all the control qubits.

**We can mitigate ancillary qubits by using gray code!**

## 3.2 Reversibility of computation

Quanutm algorithms/circuits, that use unitary operations are by definition reversible (because of the fact, that they are unitary).
If one wants to use irreversible computation we need extra space to make the computation reversible again!

**Reversibility in quantum computing is needed, so no information is lost.** This is crucial to maintain the benefits of quanutm computing and entanglement.

The additional qubits we need, to make an irreversible computation reversible are called "garbage qubits". They serve as a temporal storage, to store the necessary information to make an operation reversible. At the end of the computation the garbage qubits themselves need to be reset too (they cannot just be "thrown away"), otherwise it would be like taking a measurement, which has unwanted sideeffects on the actual computation.

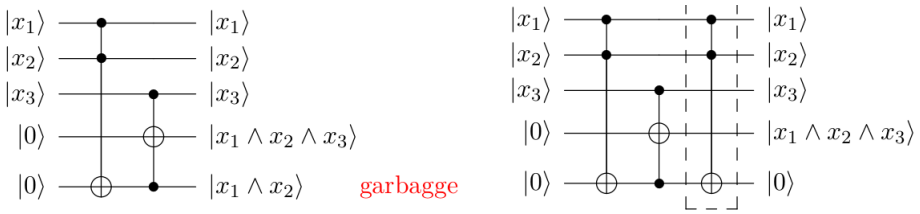



Figure 1: Example of garbage in an AND computation of $x_1 \wedge x_2 \wedge x_3$ and the uncomputation

An oracle can be made reversible, by inputing the input and the target and outputting the input and the *xor* of the initial output state and the function output itself.
So output on qubit $y$ with input $x$: $y \oplus f(x)$

## 3.3 Phase kickback

Explenation video: "Phase Kickback"

In some operations the control qubit gets changed, while the target qubits remains unchanged. It occures, if the state vector (target qubit) is an eigenvector of the operation

$U$. To measure the phase kickback effect, the control qubit must be in a superposition state (e.g. by applying a Hadamard gate first).

**Phase oracle:**

$$U_f|x\rangle|-\rangle = \left((-1)^{f(x)}|x\rangle\right)|-\rangle$$

# 4 Algorithms of quantum computers I

## 4.1 Superdense coding

Explenation video: "Quantum Computing Course: 3.1 Superdense Coding"

Superdense coding is the concept of sending two (classical) bits worth of information to another party, by using just one qubit.

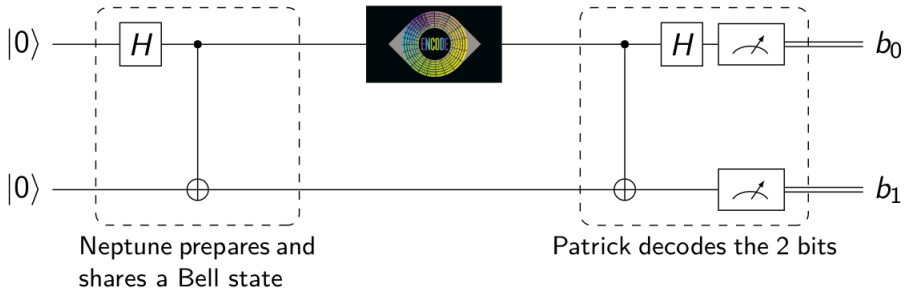



Figure 2: Circuit for Superdense Coding

1. Prepare a superposition-/bell state by applying a Hadamard gate
2. Apply some gate(s) depending on the information to send
   - **00**: Apply the $I$ operator (do nothing)
   - **01**: Apply the $Z$ operator
   - **10**: Apply the $X$ operator
   - **11**: Apply first the $X$ and then the $Z$ operator
3. To read out the encoded information the other party has to apply a $CNOT$ gate
4. After applying the $CNOT$ gate the Hadamard gate is applied aswell
5. The resulting state after the measurement corresponds to the bitstring that was sent before encoding

## 4.2 Quantum teleportation

Quantum states can be copied/teleported, but the original quantum state is destroyed this way. This is because of a rule, that quantum states cannot be cloned.

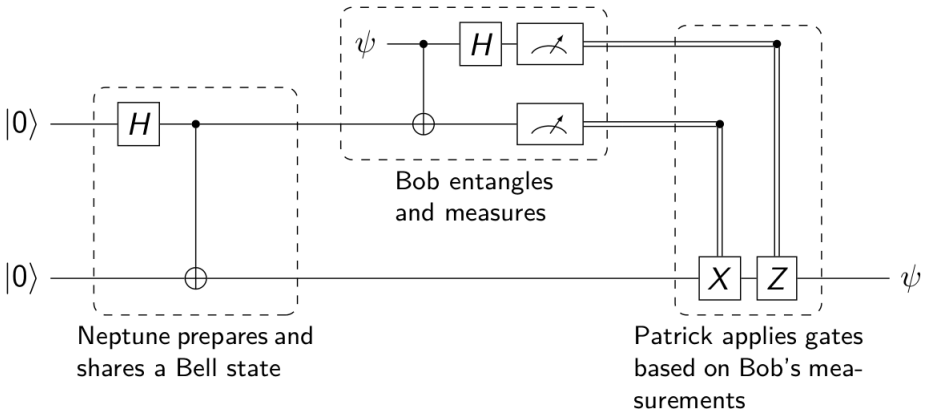



Figure 3: Circuit for Quantum Teleportation

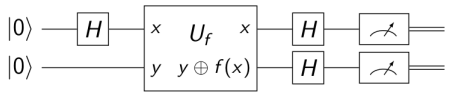
Idea:

- Sender and receiver both have an entangled pair
- The sender additionally has a superposition state which he/she wants to transfer
- By mearuring the superposition state and transmitting this measurements to the receiver, we can reconstruct the superposition state in the entangled pair

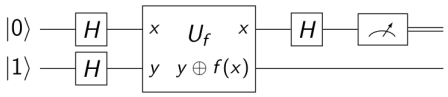
## 4.3 The Algorithm of Deutsch

Explenation video: "Deutsch's Algorithm: An Introduction to Quantum Computing Oracles"

Given an oracle function $f$, where we do not know the implementation, is $f$ constant (always returns a certain number, either 0 or 1) or balanced (it returns 0 half the time and 1 the other half of the time - e.g. $NOT$ gate)?
With a classical computer we would need **2** calls to the function to determine if it is constant or balanced. **With quantum computers we only need 1 call!**




(a) The Algorithm of Deutsch (original version)




(b) The Algorithm of Deutsch (improved version)

1. Make a superposition state from $|0\rangle$by applying a Hadamard gate to the input
$$|\Psi_1\rangle = (H|0\rangle)\,|0\rangle = |+\rangle|0\rangle = \frac{1}{\sqrt{2}}\,(|0\rangle + |1\rangle)\,|0\rangle = \frac{1}{\sqrt{2}}\,(|00\rangle + |10\rangle)$$
2. Input them to the oracle function $U_f$
$$|\Psi_2\rangle = U_f|\Psi_1\rangle = \frac{1}{\sqrt{2}}\,(|0\rangle|f(0)\rangle + |1\rangle|f(1)\rangle)$$
3. Constant or balanced?
   - If $f(0) = f(1)$, we measure halft the time $|00\rangle$ and half of the time $|01\rangle$
   - If $f(0) \neq f(1)$, we measure halft the time $|00\rangle$ and half of the time $|11\rangle$

The improved version can use the phase oracle, to determine $|\Psi_3\rangle$. We then have

- $|0\rangle|-\rangle$, if $f(0) = f(1)$
- $|1\rangle|-\rangle$, if $f(0) \neq f(1)$

## 4.4 The Algorithm of Deutsch and Jozsa

The problem is the same as for the Algorithm of Deutch, but with an arbitrary input (but still outputs 0 or 1).

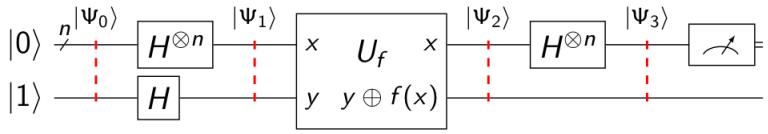



Figure 5: Circuit for the algorithm of Deutsch and Jozsa

Measurement of the first $n$ qubits is $0 \cdot ... \cdot 0$, if $f(0) = f(1)$
Measurement of the first $n$ qubits has at least one 1, if $f(0) \neq f(1)$

## 4.5 The Algorithm of Bernstein and Vazirani

The problem statement here is, to recover a secret bitstring, which is encoded in an oracle function. In a conventional computer, we can query the oracle repeatedly, which adds/*xor*s the input and the secret and outputs the result. With the quantum approach, we once again only need one oracle call. The circuit is the same as for the algorithm of

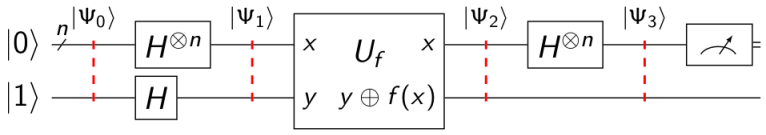



Figure 6: Circuit for the algorithm of Bernstein and Vazirani

Deutsch and Jozsa, but the oracle function $U_f$ is different. For the state $|\Psi_2\rangle$ we get:

$$
\begin{aligned}
|\Psi_2\rangle &= \frac{1}{\sqrt{2}} \left( \sum_{x \in \{0,1\}^n} (-1)^{f(x)} |x\rangle \right) |-\rangle \\
&= \frac{1}{\sqrt{2}} \sum_{x \in \{0,1\}^n} (-1)^{x \odot z} |x\rangle \\
&= H^{\otimes n} |z\rangle |-\rangle \qquad \text{for any } |z\rangle = |z_{n-1} \cdots z_0\rangle
\end{aligned}
$$

which then gives:

$$
\begin{aligned}
|\Psi_3\rangle &= H^{\otimes n} \left( H^{\otimes n} |z\rangle \right) |-\rangle \\
&= |z\rangle |-\rangle
\end{aligned}
$$

Measuring the first $n$ qubits, we can reconstruct $s$.

An optimization can be done. The Hadamard gates before and after can be omited, since the oracle only uses $CNOT$ gates to the input. This can be done, since $H \cdot CNOT \cdot H = \overline{CNOT}$ (control and target qubit get flipped).

The problem with the algorithm is, that it requires as many qubits as there are input bits. A space optimization would be to only use one output qubit, and after measureing the bit, resetting it to the initial state $|0\rangle$.

# 5 Algorithms of quantum computers II

## 5.1 Grover's Algorithm

Explenation videos: "A Visual Introduction to Grover's Algorithm and Reflections"
"Grover's Algorithm | Simplified | Quantum Computing"

The problem, that Grover's algorithm solves is, to find with high probability the unique input to an oracle function that produces a particular output value. This can for example be used in (heuristic) search problems

- Finding an item in an unstructured database

- finding a route between cities
- Finding a satisfying assignment

Let's consider the search algorithm setting.
We have a unordered list of elements of size $N = 2^n$ and we want to find the index of a specific element.
We have a function $f$ with following properties:

$$f(x) = \begin{cases} 1 & \text{if } x \text{ is the solution to the search problem} \\ 0 & \text{otherwise} \end{cases}$$

A quantum oracle $O = U_f$ performs the operation

$$U_f|x\rangle|-\rangle = (-1)^{f(x)}|x\rangle|-\rangle$$

where $f(x) = 0$ for all $0 \leq x \leq 2^n$ except for the desired $x_0$ for which $f(x_0) = 1$. We additionally have $n$ qubits in the state $|0\rangle$and one qubit in the state $|1\rangle$. Grover's

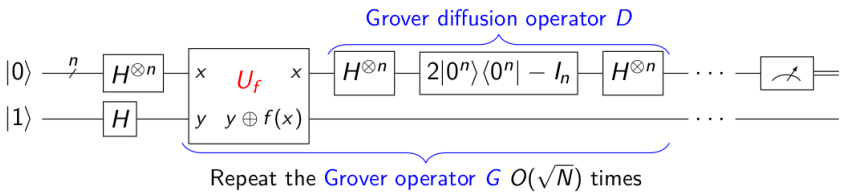



Figure 7: Circuit for Grover's algorithm

algorithm works as follows:

1. Initialize all states in a superposition (apply the Hadamard gate)
2. Apply the Grover operator $G$
   a) Apply the oracle function $U_f$, which negates the amplitude of the element $x_0$ we are looking for
      - Remember $U_f|x\rangle|-\rangle = (-1)^{f(x)}|x\rangle|-\rangle$ meaning if $f(x) = 1$ we negate the amplitude
      - For all other elements, the amplitude remains unchanged
   b) Apply the Grover diffusion operator $D$

      The matrix $2|0^n\rangle\langle 0^n| - I_n$ realizes a conditional phase shift. This means, that the negative amplitude gets flipped again to a positive amplitude (others stay positive).

$$\left(2|0^n\rangle\langle 0^n| - I_n\right)|x\rangle = \begin{cases} |x\rangle & \text{if } |x\rangle \text{ is } |0^n\rangle \\ -|x\rangle & \text{otherwise} \end{cases}$$

This operator also automatically increases the amplitude of $x_0$ and decreases the amplitudes in respect to the difference to the mean amplitude of the other elements.

This causes the amplitude peak of the element $x_0$ to be more prominent and therefore easier to detect in a measurement

3. Apply the Grover operator $G$ $\left\lfloor \frac{\pi}{4}\sqrt{2^n} \right\rfloor$ times.

   With each iteration the amplitude of $x_0$ gets more prominent

NOTE: Success probability depends on choice of iterations. Applying the Grover operator $G$ more than $\left\lfloor \frac{\pi}{4}\sqrt{2^n} \right\rfloor$ times, will decrease the amplitude of $x_0$ again and scale up the other elements. This therefore again decreases the probability of finding the correct index. The function is periodic.

- For finding a single solution, run $\left\lfloor \frac{\pi}{4}\sqrt{2^n} \right\rfloor$ times
- For finding $k$ solutions, run $\left\lfloor \frac{\pi}{4}\sqrt{\frac{2^n}{k}} \right\rfloor$ times

## 5.2 Simon's Algorithm

The problem, that Simon's algorithm solves is, that we want to find an input to a "1-1" or "2-1" mapping oracle function $f : \{0,1\}^n \mapsto \{0,1\}^n$, where the output $f(x \oplus s)$ of the input $x$ has the same result as the initial input $f(x)$. The oracle function has a secret (bit) string $s \in \{0,1\}^n$, which is *xor*'ed with the input. The problem now is, that we want to find the secret string $s$ from the oracle function.

$$f(x) = f(x \oplus s)$$

1. Input $x$ into the function $f$
2. It produces the output $f(x) = x \oplus s$
3. We input $x \oplus s$ again to $f$
4. We get $f(x \oplus s)$
5. Is $f(x) = f(x \oplus s)$?

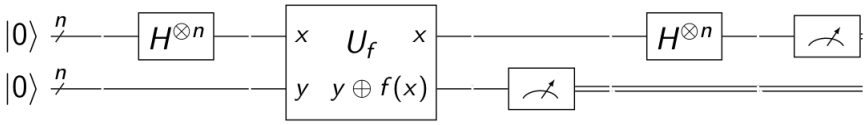



Figure 8: Circuit for Simon's algorithm

Simon's algorithm works as follows:

1. Initialize two quantum registers in a superposition (apply Hadamard gate)

   The superpositions are all possible input strings for the oracle function

2. Input one set of superpositions to the oracle function $U_f$ (input the second register)

3. The output is going to be $\frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle|f(x)\rangle$

4. Perform a measurement on the second register
   - Since all outputs are equally likely, this will give a random output $f(x)$
   - It will simultaneously collapse the input states of the first register to all inputs, which have the same output for $f(x)$!
   - This means that the input states, which will remain are $|x\rangle$ and $|x \oplus s\rangle$!

5. Apply the Hadamard gate to the first (input) register, which will again give us the states for the inputs

   $$\frac{1}{\sqrt{2^{n+1}}} \sum_{y\in\{0,1\}^n} \left[(-1)^{x\cdot y} + (-1)^{(x\oplus s)\cdot y}\right] |y\rangle$$
   $$= \frac{1}{\sqrt{2^{n+1}}} \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y} \cdot \left[1 + (-1)^{s\cdot y}\right] |y\rangle$$

6. Measure the first register (which will always give two results)
   - An all 0 state for $s \cdot y = 1$ (this will not be measured, since the amplitude of this case is 0)
   - A state for which $s \cdot y = 0$

7. Repeat the algorithm $\mathbf{n - 1}$ times to collect enough (independent) $y$'s $(y_1, y_2, ..., y_{n-1})$, such that $s \cdot y_k = 0$

8. By solving the linear equations $s \cdot y_k = 0$ we can find out $s$

Remark: In step 7. we say "independent $y$'s", this refers to the fact, that a $y$, which is a composition of (an-)other $y$ does not contribute to finding a solution. Getting such a independent $y$ has probability $\frac{1}{4}$.

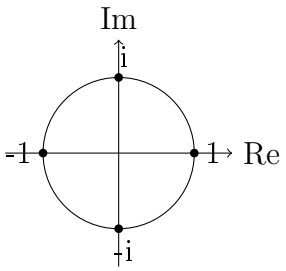
# 6 Algorithms of quantum computers III

## 6.1 Quantum Fourier transform (QFT)

A fourier transform translates a signal from the time domain (signal over time) to a frequency domain (what frequencies make up the signal). A discrete fourier transform is actually just a fourier transform of a finite, signal with discrete datapoints.

$$QFT|j\rangle = \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{\frac{2\pi i j k}{2^n}} |k\rangle$$

The above fromula creates a uniform distribution of amplitudes of the states, that differ in their phase to each other. The increasing $k$ actually tells us "how large the steps-size around the unit circle" is.




Note that we can write the QFT as a tensor product.

$$\Psi_n = \frac{(|0\rangle + e^{2\pi i 0.j_0}|1\rangle) \otimes (|0\rangle + e^{2\pi i 0.j_1 j_0}|1\rangle) \otimes \cdots \otimes (|0\rangle + e^{2\pi i 0.j_{n-1}\cdots j_0}|1\rangle)}{\sqrt{2^n}}$$

This means, that the states in $\Psi_n$ are not entangled! So measuring qubits does not cause other qubits to collapse. If we measure the qubits, we actually get the reverse order, in which the bits were provided. To fix this we can apply $SWAP$ gates and the end to reorder the states before measuring. A simpler approach is also to just relabel the qubits at the end.

## 6.2 Quantum Phase Estimation (QPE)

Quantum phase estimation (QPE) is used to estimate the phase of a given eigenvector corresponding to a given (yet unknown) eigenvalue of a unitary operator. By finding the phase $\varphi$ with high probability, we can determine the eigenvalue.

$$U|u\rangle = e^{i\varphi}|u\rangle$$

As can be seen in figure 9 the algorithm works as follows:

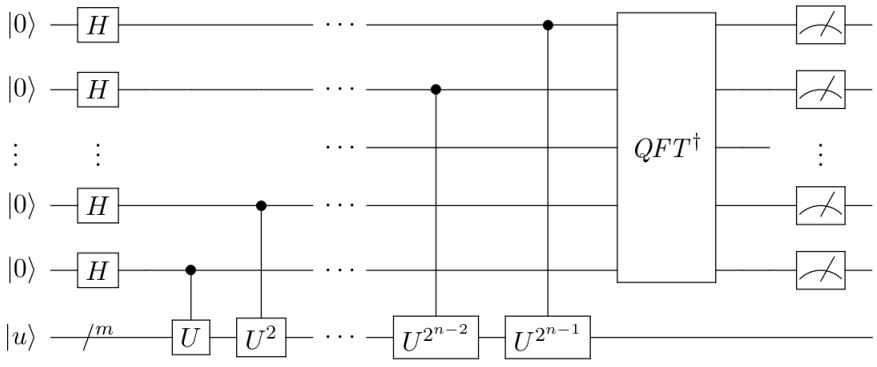



Figure 9: Circuit for quantum phase estimation

1. Initialize $n$ qubits with $|0\rangle$and use $m$ qubits to describe $|u\rangle$ (the eigenvector we want the eigenvalue of)

2. Create a superposition of states in the first register for the phase estimation

3. Apply the controlled unitary operator $CU$ a power of $2^{n-1}$ times. So $CU^{2^{n-1}}$

   Note, that because $|u\rangle$ is an eigenvector of $U$, phase kickback occures and we apply a relative phase to the control qubit (first register)

4. Use the inverse QFT ($QFT^\dagger$) of the first register to get the phase estimation.

5. There can be an assiciated error in the accuracy of the measurement with $0 \leq |\delta| \leq \frac{1}{2^{n+1}}$

   If $\delta = 0$ the estimation is exact

# 7 Algorithms of quantum computers IV

## 7.1 Shor's Algorithm

Explenation videos: "How Quantum Computers Break The Internet... Starting Now" (Basic)
"11: Algorithms for order-finding and factoring" (In-depth)

The problem, that Shor's algorithm solves is, Given an integer $n > 2$, find its prime factors. The computation for classical computers is hard or even "provably infeasible" for big numbers. The RSA encryption depends on exactly this principle and so does the Diffie-Hellman key exchange.

### 7.1.1 Idea

We want to find the prime factors $p$, $q$ of a number $N$ ($p \cdot q = N$):

1. Guess a number $g$, that does not share factors with $N$

2. Raise $g$ to some numnber $r$, s.t. $g^r = m \cdot N + 1$

   Then to find $r$ we calculate $g^x \mod N = 1$ for $x \in \{1, 2, ...\}$.

   One can rewrite the equation to: $\underbrace{\left(g^{r/2} + 1\right)}_{a \cdot \mathbf{p}} \cdot \underbrace{\left(g^{r/2} - 1\right)}_{b \cdot \mathbf{q}} = m \cdot N$

   Note, that since we have $g^{r/2}$ in the equation, $r$ must be an even number. If $r$ is odd, we have to find another $r$

3. Use Euclids algorithm to find $gcd(g^{r/2} \pm 1, N) = p$

4. Then $\frac{N}{p} = q$

### 7.1.2 Quantum implementation

The key computation, where we use quantum properties to speed up the computation is in finding the number $r$, s.t. $g^r = m \cdot N + 1$.

If we have a superposition of $g^x$ we can calculate $g^x \mod N$. The remainder of the calculation repeats periodically, meaning all following numbers, which share the same remainder are spaced out by $r$. Then make a measurement and get a random $x$ and remainder. Notice, that now also the superposition collapses to only include $x$, which share the same remainder.

1. Prepare two qubit registers
   - The first contains a superposition of all numbers $x$ for $g^x$
   - The second is initialized to $|0\rangle$
2. Devide $\frac{g^x}{N}$ and store the remainder in the second register

   Notice, that with this we have entangled the first and the second register
3. Measure the second register to obtain a random remainder

   This collapses the first register to includ only exponends that share this same remainder
4. Apply the QFT to get $\frac{1}{r}$

   The measurement has to be repeated multiple time to eventually derive $\frac{1}{r}$, since when measuring we will always get a random multiple or the fraction $c \cdot \frac{1}{r}$

   We can then use the Quanutm Phase Estimation (QPE) to approximate $\frac{1}{r}$

   Invert the fraction and we have $r$

   Now we know $\mathbf{g^r = m \cdot N + 1}$
5. If $r$ is even we have turned our initial bad guess $g$ into a better guess $g^r$, which likely shares factors with $N$ (aslong as $(g^r \pm 1)$ are not multiples of $N$)

After this we can continue classically with Euclids algorithm and find $p$ and $q$.

NOTE: Choose a $q$ (the number of qubits of the first register), s.t. $q = 2^l$ and $n^2 \leq q \leq 2n^2$ to increase the chances of determining a unique $r$
The more qubits, the clearer we can derive $r$ from the measurement(?). The upper bound is just a practicality, since it is physically hard to have such many gates.