Summer of Code

Quantum Computing: From Theory to Practice

Aakash Tarang

IIT BOMBAY

June 2025

Contents

1 Linear Algebra **1**

1.1 Vector Space $\mathbb{C}^n$ . . . 1
1.2 Bases and Linear Independence . . . 1
1.3 Linear Operators and Matrices . . . 2
1.4 The Pauli Matrices . . . 2
1.5 Inner Product . . . 2
1.5.1 Definition . . . 2
1.5.2 Properties . . . 3
1.5.3 Important Concepts . . . 3
1.6 Hilbert Space . . . 3
1.7 Gram-Schmidt Orthogonalization . . . 3
1.8 Outer Product . . . 4
1.9 Special Operators . . . 4
1.9.1 Hermitian Operators . . . 4
1.9.2 Unitary Operators . . . 4
1.9.3 Positive Operators . . . 4
1.10 Eigenvectors and Eigenvalues . . . 4
1.11 Hermitian and Unitary Operators . . . 4
1.12 Projectors . . . 5
1.13 Tensor Products . . . 5
1.13.1 Tensor Product Examples and Notation . . . 5
1.14 Operator Functions . . . 6
1.15 Commutators and Anti-commutators . . . 6
1.16 Matrix Decompositions . . . 7
1.17 Matrix Exponentials . . . 7
1.18 Trace . . . 7

2 Quantum Measurement Theory **9**

2.1 The Standard Lore and Its Limitations . . . 9
2.1.1 Traditional Quantum Measurement Framework . . . 9
2.1.2 Limitations of the Standard Framework . . . 9
2.1.3 Table Explanation . . . 10

3 Introduction to Quantum Computing **11**

3.1 The Quantum Advantage . . . 11
3.2 Historical Milestones . . . 11
3.3 A Practical Example: **Superdense Coding** . . . 11
3.3.1 Initial Setup . . . 11
3.3.2 Creating and Decoding Entangled Pairs . . . 12
3.3.3 Sending a Full Message . . . 12
3.3.4 Verification using AerSimulator . . . 13

3.3.5 Simulating Realistic Noise . . . 13
3.3.6 Conclusion . . . 14

4 Qubits and Quantum Gates **15**
4.1 Qubit States . . . 15
4.2 Superposition Principle . . . 15
4.3 Bloch Sphere Representation . . . 15
4.4 Single-Qubit Gates . . . 15
4.5 Quantum Dynamics . . . 16
4.6 Key Properties . . . 16

5 Quantum Search **17**
5.1 Grover's Algorithm . . . 17
5.1.1 The Algorithm Steps . . . 17
5.1.2 Performance and Measurement . . . 18

6 Quantum Fourier Transform **19**
6.1 Quantum Fourier Transform (QFT) Summary . . . 19
6.1.1 Computational Basis . . . 19
6.1.2 QFT Definition . . . 19
6.1.3 Physical Interpretation . . . 19
6.1.4 QFT as Linear Transformation . . . 20
6.1.5 Single Qubit Example ($n = 1$) . . . 20
6.1.6 Key Properties . . . 20
6.1.7 Realization Using Quantum Gates . . . 20

CHAPTER 1

Linear Algebra

1.1 Vector Space $\mathbb{C}^n$

A vector space of complex numbers is a set of n-tuples of complex numbers, whose elements are called vectors, which satisfy certain vector space axioms. The vectors are indicated using column matrix notation:

$$\begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix}$$

In $\mathbb{C}^n$, addition for vectors is defined as:

$$\begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix} + \begin{bmatrix} z'_1 \\ \vdots \\ z'_n \end{bmatrix} = \begin{bmatrix} z_1 + z'_1 \\ \vdots \\ z_n + z'_n \end{bmatrix}$$

Multiplication is defined as:

$$z \begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix} = \begin{bmatrix} zz_1 \\ \vdots \\ zz_n \end{bmatrix}$$

The standard quantum mechanical notation for representing a vector in a vector space is $|\psi\rangle$, where ψ is a label for the vector, and the entire expression is called a ket.

The zero vector is represented by 0 (the ket notation is not used for the zero vector).

A vector subspace of a vector space V is a subset W of V such that W is also a vector space.

1.2 Bases and Linear Independence

A spanning set for a vector space is a set of vectors $\{|v_1\rangle, \ldots, |v_n\rangle\}$ such that any vector $|v\rangle$ in the vector space can be written as a linear combination $|v\rangle = \sum_i a_i |v_i\rangle$ of vectors in that set.

A set of non-zero vectors $\{|v_1\rangle, \ldots, |v_n\rangle\}$ are linearly dependent if there exists a set of complex numbers $a_1, \ldots, a_n$ with $a_i \neq 0$ for at least one value of i, such that:

$$a_1 |v_1\rangle + a_2 |v_2\rangle + \cdots + a_n |v_n\rangle = 0$$

A set of vectors is linearly independent if it is not linearly dependent. Any two sets of linearly independent vectors which span a vector space V contain the same number of elements. Such a set is called a basis and the cardinality of a basis is called its dimension.

1.3 Linear Operators and Matrices

A linear operator between vector spaces V and W is defined to be any function $A : V \to W$ which is linear in its inputs:

$$A\left(\sum_i a_i |v_i\rangle\right) = \sum_i a_i A(|v_i\rangle)$$

An important linear operator on any vector space V is the identity operator I_V defined by $I_V |v\rangle = |v\rangle$ for all vectors $|v\rangle$. Another important linear operator is the zero operator denoted by 0 which maps all vectors to the zero vector, $0 |v\rangle = 0$.

Once the action of a linear operator A on a basis is specified, the action of A is completely determined on all inputs.

Suppose V, W, and X are vector spaces, and $A : V \to W$ and $B : W \to X$ are linear operators. Then BA denotes the composition of B with A, defined as $BA |v\rangle = B(A(|v\rangle))$.

Linear operators can be understood in terms of their matrix representations. Matrices can be regarded as linear operators and linear operators can be represented as matrices, thereby making the two completely equivalent. Suppose $A : V \to W$ is a linear operator between vector spaces V and W. Suppose $\{|v_1\rangle, \ldots, |v_m\rangle\}$ is a basis for V and $\{|w_1\rangle, \ldots, |w_n\rangle\}$ is a basis for W. Then for each j in the range $1, \ldots, m$, there exist complex numbers A_{1j} through A_{nj} such that:

$$A |v_j\rangle = \sum_i A_{ij} |w_i\rangle$$

The matrix whose entries are the values A_{ij} is said to form a matrix representation of the operator A.

1.4 The Pauli Matrices

Four useful 2×2 matrices used occasionally are the Pauli matrices:

$$\sigma_0 = I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \sigma_1 = \sigma_x = X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$\sigma_2 = \sigma_y = Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad \sigma_3 = \sigma_z = Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

1.5 Inner Product

An inner product is a function that takes two vectors $|v\rangle$ and $|w\rangle$ from a vector space and produces a complex number, denoted $\langle v|w\rangle$.

1.5.1 Definition

In quantum mechanics, the inner product is typically defined as:

$$\langle v|w\rangle = \int v^*(x) w(x) t(x) dx$$

where $t(x)$ is a weight factor (e.g., $t(x) = 1$ for Cartesian coordinates, $t(x) = r^2 \sin\theta$ for spherical coordinates).

1.5.2 Properties

- **Linearity in the second argument**:

$$\langle v|aw_1 + bw_2\rangle = a\langle v|w_1\rangle + b\langle v|w_2\rangle$$

- **Conjugate symmetry**:

$$\langle v|w\rangle = \langle w|v\rangle^*$$

- **Positive-definiteness**:

$$\langle v|v\rangle \geq 0 \quad \text{with equality iff } |v\rangle = 0$$

- **Orthogonality**:

$$\langle v|w\rangle = 0 \quad \text{if } |v\rangle \text{ and } |w\rangle \text{ are orthogonal}$$

1.5.3 Important Concepts

- **Norm**: $\| |v\rangle \| = \sqrt{\langle v|v\rangle}$
- **Cauchy-Schwarz inequality**:

$$|\langle v|w\rangle|^2 \leq \langle v|v\rangle \langle w|w\rangle$$

- **Triangle inequality**:

$$\| |v\rangle + |w\rangle \| \leq \| |v\rangle \| + \| |w\rangle \|$$

- **Projection**:

$$\text{proj}_w(v) = \frac{\langle w|v\rangle}{\langle w|w\rangle} |w\rangle$$

1.6 Hilbert Space

In quantum computation, a Hilbert space is an inner product space with additional completeness properties. Key features:

- $\langle v|$ is the dual vector (bra) to $|v\rangle$ (ket)
- Vectors $|w\rangle$ and $|v\rangle$ are orthogonal if $\langle w|v\rangle = 0$
- A set $\{|i\rangle\}$ is orthonormal if $\langle i|j\rangle = \delta_{ij}$

1.7 Gram-Schmidt Orthogonalization

Given a basis $\{|w_1\rangle, ..., |w_d\rangle\}$, we can construct an orthonormal basis $\{|v_1\rangle, ..., |v_d\rangle\}$:

$$|v_1\rangle = \frac{|w_1\rangle}{\| |w_1\rangle \|}, \quad |v_{k+1}\rangle = \frac{|w_{k+1}\rangle - \sum_{i=1}^{k} \langle v_i|w_{k+1}\rangle |v_i\rangle}{\| |w_{k+1}\rangle - \sum_{i=1}^{k} \langle v_i|w_{k+1}\rangle |v_i\rangle \|}$$

1.8 Outer Product

The outer product $|w\rangle\langle v|$ is a linear operator defined by:

$$(|w\rangle\langle v|)(|v'\rangle) = \langle v|v'\rangle|w\rangle$$

Key applications:

- **Projection operator**: $P = |v\rangle\langle v|$
- **Completeness relation**: $\sum_i |i\rangle\langle i| = I$
- **Operator representation**: $A = \sum_{ij} \langle w_j|A|v_i\rangle |w_j\rangle\langle v_i|$

1.9 Special Operators

1.9.1 Hermitian Operators

An operator A is Hermitian if $A^\dagger = A$, where:

$$\langle v|A|w\rangle = \langle A^\dagger v|w\rangle = \langle w|A|v\rangle^*$$

1.9.2 Unitary Operators

An operator U is unitary if $UU^\dagger = I$. They preserve inner products:

$$\langle Uv|Uw\rangle = \langle v|w\rangle$$

1.9.3 Positive Operators

An operator A is positive if $\langle v|A|v\rangle \geq 0$ for all $|v\rangle$.

1.10 Eigenvectors and Eigenvalues

Let A be a linear operator on vector space V. A non-zero vector $|v\rangle$ is an **eigenvector** of A with **eigenvalue** $v \in \mathbb{C}$ if:

$$A|v\rangle = v|v\rangle$$

The **eigenspace** for eigenvalue v is the subspace of all corresponding eigenvectors.

An operator is **diagonalizable** if it has a **diagonal representation**:

$$A = \sum_i \lambda_i |i\rangle\langle i|$$

where $\{|i\rangle\}$ form an orthonormal eigenbasis with eigenvalues λ_i.

1.11 Hermitian and Unitary Operators

The **adjoint** $A^\dagger$ satisfies:

$$(|v\rangle, A|w\rangle) = (A^\dagger|v\rangle, |w\rangle)$$

In matrix form, $A^\dagger = (A^*)^T$.

Key operator classes:

- **Hermitian**: $A^\dagger = A$
- **Unitary**: $UU^\dagger = I$ (preserves inner products)
- **Normal**: $AA^\dagger = A^\dagger A$ (diagonalizable)
- **Positive**: $\langle v|A|v\rangle \geq 0$ for all $|v\rangle$

1.12 Projectors

For subspace $W \subseteq V$ with orthonormal basis $\{|i\rangle\}_{i=1}^{k}$, the **projector** is:

$$P = \sum_{i=1}^{k} |i\rangle \langle i|$$

Properties:

- $P^\dagger = P$ (Hermitian)
- $P^2 = P$ (idempotent)
- $Q = I - P$ is the orthogonal complement

1.13 Tensor Products

For vector spaces V and W, the tensor product $V \otimes W$ has:

- Elements: linear combinations of $|v\rangle \otimes |w\rangle$
- Basis: $\{|i\rangle \otimes |j\rangle\}$ for bases $\{|i\rangle\}$, $\{|j\rangle\}$
- Operator action: $(A \otimes B)(|v\rangle \otimes |w\rangle) = A|v\rangle \otimes B|w\rangle$

1.13.1 Tensor Product Examples and Notation

The tensor product combines vectors from different vector spaces, essential for constructing multi-qubit systems in quantum computing.

Why Tensor Products? (Enlarging the Hilbert Space)

- **Purpose**:
  - A single qubit lives in a 2D Hilbert space ($\mathbb{C}^2$)
  - For n qubits, the state space must describe all 2^n possible combinations
  - The tensor product $\mathbb{C}^2 \otimes \mathbb{C}^2 \otimes \cdots \otimes \mathbb{C}^2 = \mathbb{C}^{2^n}$ gives the correct dimension
- **Physical Interpretation**:
  - Combines independent systems: $|0\rangle \otimes |1\rangle = |01\rangle$
  - Emerges naturally from quantum mechanics postulates
  - Physically achieved through qubit coupling (e.g., trapped ions, superconducting circuits)

Notation

- $|a\rangle \otimes |b\rangle$ - Explicit tensor product
- $|ab\rangle$ or $|a, b\rangle$ - Common shorthand
- $|abc\rangle \equiv |a\rangle \otimes |b\rangle \otimes |c\rangle$ - Multi-qubit states
- $|00010001111\rangle$ - 11-qubit state (one-hot encoded)

One-Hot Encoded States

States like $|00010001111\rangle$ represent:

- Binary string interpretation: $00010001111_2 = 271_{10}$
- One-hot vector with 1 at position 271 (0-indexed)
- In $\mathbb{C}^{2048}$ space (since $2^{11} = 2048$)
- Computational basis state for 11-qubit systems

Example: Two-Qubit System

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

$$|01\rangle = \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\1\\0\\0\end{pmatrix}$$

$$|10\rangle = \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\0\\1\\0\end{pmatrix}$$

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

Key Properties

- **Dimension Growth**: $\dim(V^{\otimes n}) = (\dim V)^n$
- **Basis Construction**: $\{|i_1\rangle \otimes \cdots \otimes |i_n\rangle\}$ forms basis for $\mathbb{C}^{2^n}$
- **Physical Realization**: Implemented through quantum gates acting on multiple qubits (e.g., CNOT entangles qubits)

1.14 Operator Functions

For normal operator $A = \sum_a a\,|a\rangle\langle a|$ and function f:

$$f(A) = \sum_a f(a)\,|a\rangle\langle a|$$

1.15 Commutators and Anti-commutators

- **Commutator**: $[A, B] = AB - BA$
- **Anti-commutator**: $\{A, B\} = AB + BA$

Simultaneous Diagonalization: Hermitian A and B commute iff they share an eigenbasis.

1.16 Matrix Decompositions

- **Polar**: $A = UJ = KU$ with U unitary, J, K positive
- **SVD**: $A = UDV$ with U, V unitary, D diagonal
- **Spectral**: $A = \sum_i \lambda_i |i\rangle \langle i|$ for normal A

1.17 Matrix Exponentials

For Hermitian H:

$$e^{i\gamma H} = \sum_{n=0}^{\infty} \frac{(i\gamma H)^n}{n!}$$

If $B^2 = I$ (involutory):

$$e^{i\gamma B} = \cos(\gamma) I + i \sin(\gamma) B$$

1.18 Trace

The trace satisfies:

- $\operatorname{tr}(AB) = \operatorname{tr}(BA)$ (cyclic)
- $\operatorname{tr}(A + B) = \operatorname{tr}(A) + \operatorname{tr}(B)$ (linear)
- $\operatorname{tr}\left(UAU^\dagger\right) = \operatorname{tr}(A)$ (unitary invariance)
- $\operatorname{tr}(A |\psi\rangle \langle\psi|) = \langle\psi| A |\psi\rangle$

CHAPTER 2

Quantum Measurement Theory

2.1 The Standard Lore and Its Limitations

2.1.1 Traditional Quantum Measurement Framework

In conventional quantum mechanics courses, we learn the following postulates about quantum measurement:

- The **state** of a system is represented by a vector $|\psi\rangle$ in a Hilbert space $\mathcal{H}$
- **Observables** correspond to Hermitian operators $A = A^\dagger$ on $\mathcal{H}$
- The **spectral theorem** guarantees diagonalization: $A = \sum_a a \, |a\rangle \langle a|$
- Measurement yields outcome a with probability $P(a) = |\langle a|\psi\rangle|^2$
- **State collapse** occurs: $|\psi\rangle \to |a\rangle$ after measurement

Table 2.1: The standard quantum measurement rules

	Pure states	**General states**
State	$\|\psi\rangle \in \mathcal{H}$	$\rho = \sum_i p_i \, \|i\rangle \langle i\|$
Measurement outcomes	Eigenvalues a of $A = A^\dagger$	Same
Probabilities	$P(a) = \|\langle a\|\psi\rangle\|^2$	$P(a) = \text{tr}(\|a\rangle \langle a\| \, \rho)$
State update	$\|\psi\rangle \to \|a\rangle$	$\rho \to \|a\rangle \langle a\|$

2.1.2 Limitations of the Standard Framework

The traditional measurement paradigm has several notable limitations:

- **Measurement implementation unspecified**: The rules describe outcomes but provide no mechanism for how measurements are physically realized
- **Measurement problem**: The framework doesn't explain the transition from quantum superposition to definite measurement outcomes
- **Narrow applicability**: While valid for projective measurements, many realistic measurement scenarios don't fit this idealized picture
- **State collapse postulate**: The instantaneous nature of state collapse remains philosophically problematic and experimentally untestable

2.1.3 Table Explanation

Table 2.1 compares the measurement rules for pure states versus general (mixed) states:

- The **pure state** column describes systems with definite state vectors
- The **general state** column handles statistical mixtures using density matrices ρ
- Both cases share the same measurement outcomes (eigenvalues of A)
- Probability calculations differ: pure states use inner products while mixed states require the trace operation
- State update rules maintain consistency between the two representations

This traditional framework, while mathematically elegant, leaves open fundamental questions about the nature of measurement in quantum mechanics.

CHAPTER 3

Introduction to Quantum Computing

3.1 The Quantum Advantage

Quantum computing represents a paradigm shift from classical computing by leveraging quantum phenomena such as superposition and entanglement. These properties allow quantum computers to solve certain problems exponentially faster than classical computers, with notable applications in cryptography, optimization, and quantum simulation.

3.2 Historical Milestones

The concept of quantum computing was first proposed by Richard Feynman in 1982. Since then, developments such as Shor's algorithm (1994) and the creation of actual quantum hardware have rapidly advanced the field. Tools like IBM's Qiskit have made quantum programming accessible to the public, enabling researchers and students alike to experiment on real or simulated quantum devices.

3.3 A Practical Example: Superdense Coding

In this section, I explore superdense coding — a powerful protocol that allows two classical bits to be transmitted using just one qubit, given a shared entangled state. This is achieved by exploiting quantum entanglement and specific unitary transformations to encode information.

3.3.1 Initial Setup

I began by writing helper functions in Python using Qiskit to convert strings to binary and back, to allow us to transmit messages via superdense coding.

```
def wordtobinary(word):
   return ''.join(format(ord(c), '08b') for c in word)

def binarytoword(binarystr):
   padded = binarystr.ljust((len(binarystr) + 7) // 8 * 8, '0')
   return ''.join(chr(int(padded[i:i+8], 2)) for i in range(0,
      len(padded), 8))
```

Listing 3.1: Word-Binary Conversion Helpers

3.3.2 Creating and Decoding Entangled Pairs

Using Qiskit, I defined a function to simulate the entire superdense coding protocol on an ideal quantum simulator using statevectors. Here, we create an entangled pair, encode a 2-bit message, and decode it back on the receiver side.

```
from qiskit import QuantumCircuit
from qiskit.quantuminfo import Statevector

def send1(bits):
   qc = QuantumCircuit(2)
   qc.h(0)
   qc.cx(0, 1)
   if bits == '01':
      qc.x(0)
   elif bits == '10':
      qc.z(0)
   elif bits == '11':
      qc.z(0)
      qc.x(0)
   qc.cx(0, 1)
   qc.h(0)
   return qc
```

Listing 3.2: Send and Decode using Statevector

I verified that the original 2-bit strings were correctly recovered using Qiskit's `Statevector` object.

```
def decodestatevector(sv):
   vec = sv.data
   for i, amp in enumerate(vec):
      if abs(amp) ¿ 1e-6:
         return format(i, '02b')[::-1]
```

Listing 3.3: Decoding Statevector Result

3.3.3 Sending a Full Message

I extended the protocol to send full messages like "hello" and "Aakash". Each character was converted to binary, broken into 2-bit chunks, transmitted using superdense coding, and decoded.

```
message = "hello"
binary = wordtobinary(message)
bitslist = [binary[i:i+2] for i in range(0, len(binary), 2)]

receivedbits = []
for bits in bitslist:
   qc = send1(bits)
   sv = Statevector.fromlabel('00').evolve(qc)
   receivedbits.append(decodestatevector(sv))

finalbinary = ''.join(receivedbits)
print(binarytoword(finalbinary))
```

Listing 3.4: Full Message Transmission using Ideal Simulation

3.3.4 Verification using AerSimulator

To validate the correctness of the protocol, I implemented measurement-based verification using Qiskit's `AerSimulator`. I tested all four input cases and verified perfect transmission.

```
def send1measured(bits):
   qc = QuantumCircuit(2, 2)
   qc.h(0)
   qc.cx(0, 1)
   if bits == '01':
      qc.x(0)
   elif bits == '10':
      qc.z(0)
   elif bits == '11':
      qc.z(0)
      qc.x(0)
   qc.cx(0, 1)
   qc.h(0)
   qc.measure(0, 0)
   qc.measure(1, 1)
   return qc
```

Listing 3.5: Send and Measure Circuit

The results confirmed ideal transmission:

```
-'00': -'00': 1024, '01': -'10': 1024, '10': -'01': 1024, '11': -'11': 10
```

3.3.5 Simulating Realistic Noise

To mimic a real device, I simulated noisy conditions using a previously saved IBM backend noise model. I loaded the noise model and evaluated each 2-bit message under 1024 repeated trials to get the most probable result.

```
from qiskitaer import AerSimulator
simnoise = AerSimulator(noisemodel=noisemodel, basisgates=
   basisgates)

message = "Aakash"
binary = wordtobinary(message)
bitpairs = [binary[i:i+2] for i in range(0, len(binary), 2)]
received = []

for bits in bitpairs:
   qc = send1measured(bits)
   tqc = transpile(qc, simnoise)
   result = simnoise.run(tqc, shots=1024).result()
   counts = result.getcounts()
   bitstring = max(counts.items(), key=lambda x: x[1])[0]
   received.append(bitstring[::-1])

finalbinary = ''.join(received)
print(binarytoword(finalbinary)) # Should print "Aakash"
```

Listing 3.6: Transmission with Noisy Simulator

Despite the presence of noise, the protocol was able to recover the message accurately by using majority voting over multiple trials.

3.3.6 Conclusion

Through this experiment, I was able to demonstrate superdense coding not only in theory but also through both ideal and noisy simulations. The use of Qiskit allowed for hands-on validation of entanglement-based communication, laying the foundation for future exploration in quantum networking.

CHAPTER 4

Qubits and Quantum Gates

4.1 Qubit States

A qubit is the quantum analog of a classical bit, with two basis states:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

The general state of a qubit is a superposition:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \quad \text{where} \quad |\alpha|^2 + |\beta|^2 = 1$$

4.2 Superposition Principle

Unlike classical bits, qubits can exist in superposition states that are linear combinations of the basis states:

- Enables parallel computation on multiple states
- Measurement collapses the state: $P(|x\rangle) = |\,\langle x|\psi\rangle\,|^2$
- Global phase ($e^{i\phi}$) is physically unobservable

4.3 Bloch Sphere Representation

Any qubit state can be visualized on the Bloch sphere:

$$|\psi\rangle = \cos\frac{\theta}{2}\,|0\rangle + e^{i\phi}\sin\frac{\theta}{2}\,|1\rangle$$

- $|0\rangle$ at north pole, $|1\rangle$ at south pole
- θ: polar angle, ϕ: azimuthal angle

4.4 Single-Qubit Gates

Unitary operators acting on qubits:

- **Pauli gates**:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},\ Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix},\ Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

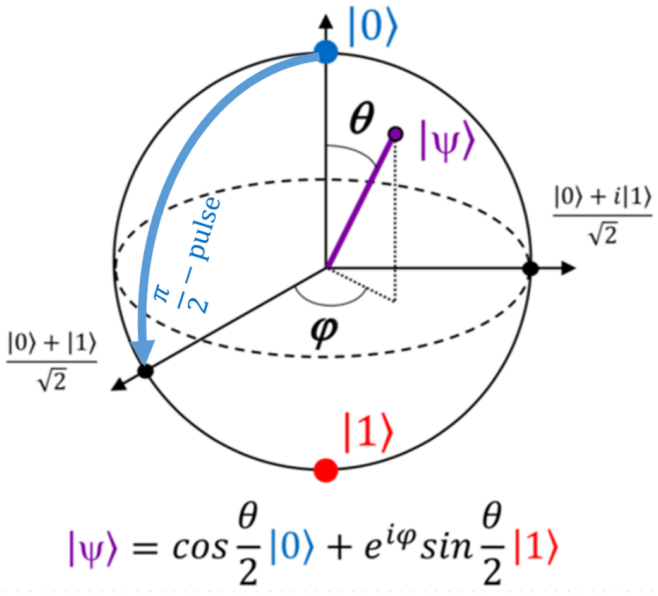

Figure 4.1: Block Sphere Representation

- **Hadamard** (creates superposition):

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

- **Phase gate**:

$$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$$

4.5 Quantum Dynamics

- Closed system evolution: $|\psi(t)\rangle = U(t)\,|\psi(0)\rangle$
- Governed by Schrödinger equation: $i\hbar \frac{d}{dt}\,|\psi\rangle = H\,|\psi\rangle$
- Hamiltonian generates unitary evolution: $U = e^{-iHt/\hbar}$

4.6 Key Properties

- **No-cloning**: Cannot copy unknown quantum states
- **Entanglement**: Qubits can exhibit non-classical correlations
- **Measurement**: Projective and irreversible

CHAPTER 5

Quantum Search

5.1 Grover's Algorithm

Grover's algorithm is a quantum search algorithm that provides a significant speedup over its classical counterparts for unstructured search problems. Developed by Lov Grover in 1996, it can find a specific entry in an unsorted database of size N in approximately $O(\sqrt{N})$ time, whereas a classical algorithm would require $O(N)$ time on average.

The problem can be framed as follows: given a function $f(x)$ which is a "black box" or an "oracle", we want to find the input x_0 for which $f(x_0) = 1$, where for all other inputs x, $f(x) = 0$. The input x_0 is often called the "marked" or "winning" state, denoted as $|w\rangle$.

5.1.1 The Algorithm Steps

Grover's algorithm consists of three main stages: initialization, iterative amplification, and measurement. Let's assume our system has n qubits, so the total number of states is $N = 2^n$.

Initialization

The algorithm begins by preparing the system in a uniform superposition of all possible states. This is achieved by applying a Hadamard gate (H) to each of the n qubits, which are initially in the state $|0\rangle$.

$$|s\rangle = H^{\otimes n} |0\rangle^{\otimes n} = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$$

Here, $|s\rangle$ is the equal superposition state. Every possible state $|x\rangle$ has an equal amplitude of $\frac{1}{\sqrt{N}}$.

The Grover Iteration

This is the core of the algorithm and is repeated approximately $\frac{\pi}{4}\sqrt{N}$ times. Each iteration consists of two main operations: the oracle and the Grover diffusion operator.

1. The Oracle (U_w): The oracle is a unitary operator that recognizes the marked state $|w\rangle$ without revealing it. It applies a phase shift of -1 to the marked state and leaves all other states unchanged.

$$U_w |x\rangle = \begin{cases} -|x\rangle & \text{if } x = w \\ |x\rangle & \text{if } x \neq w \end{cases}$$

This can be written more compactly as $U_w = I - 2|w\rangle\langle w|$, where I is the identity matrix. Applying the oracle to our superposition state $|s\rangle$ flips the sign of the amplitude of the marked state $|w\rangle$.

2. The Grover Diffusion Operator (U_s): The second step is to apply the Grover diffusion operator, U_s, which performs an "inversion about the mean." This operation amplifies the amplitude of the marked state while decreasing the amplitudes of all other states. The operator is defined as:

$$U_s = 2 |s\rangle \langle s| - I$$

where $|s\rangle$ is the uniform superposition state we started with. Geometrically, this operation reflects each state's amplitude about the average amplitude.

One full Grover iteration is the application of these two operators: $G = U_s U_w$.

5.1.2 Performance and Measurement

After applying the Grover iteration k times, the state of the system is $|\psi_k\rangle = (U_s U_w)^k |s\rangle$. The optimal number of iterations is $k \approx \frac{\pi}{4}\sqrt{N}$. After this many iterations, the amplitude of the marked state $|w\rangle$ is very close to 1, while the amplitudes of all other states are close to 0.

Finally, we perform a measurement on the qubits in the computational basis. Due to the high amplitude of the marked state, the probability of measuring $|w\rangle$ is nearly 100%. This reveals the solution to our search problem with high probability.

CHAPTER 6

Quantum Fourier Transform

6.1 Quantum Fourier Transform (QFT) Summary

6.1.1 Computational Basis

- For a single qubit: $\{|0\rangle, |1\rangle\}$
- For n qubits: $\{|0\rangle, |1\rangle, \ldots, |N-1\rangle\}$ where $N = 2^n$
- Standard basis for initialization and measurement

6.1.2 QFT Definition

The QFT transforms a computational basis state $|j\rangle$ to:

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

where:

- $N = 2^n$ (dimension of state space)
- $e^{2\pi i jk/N}$ are roots of unity encoding phase information

6.1.3 Physical Interpretation

The QFT operation has dual interpretations:

- **Active Transformation:**
  - Physically changes the quantum state
  - Implemented through quantum gates (Hadamard + controlled rotations)
  - Transforms $|\psi\rangle$ to $|\psi'\rangle = U_{\text{QFT}}|\psi\rangle$
  - Measurement outcomes differ between $|\psi\rangle$ and $|\psi'\rangle$
- **Passive Transformation:**
  - Represents a change of basis in the mathematical description
  - The amplitudes in $|\psi'\rangle$ (computational basis) equal the amplitudes $|\psi\rangle$ would have in the Fourier basis
  - Analogous to rotating coordinate systems while keeping the vector fixed

Analogy: Consider a vector pointing North:

- Active: Rotate the vector to point East (changes physical state)
- Passive: Rotate your map so East points North (changes description)

6.1.4 QFT as Linear Transformation

- Unitary transformation between computational basis and Fourier basis
- Matrix representation ($N \times N$):

$$(U_{\text{QFT}})_{k,j} = \frac{1}{\sqrt{N}} e^{2\pi i jk/N}$$

- For fixed n, this is a constant matrix

6.1.5 Single Qubit Example ($n = 1$)

$$U_{\text{QFT}} = H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

- Applies Hadamard transform
- $\text{QFT}|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$
- $\text{QFT}|1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$

6.1.6 Key Properties

- Exponential speedup over classical FFT ($O(n^2)$ vs $O(N \log N)$)
- Basis change reveals phase/frequency information
- Core component of many quantum algorithms (e.g., Shor's algorithm)
- Physical implementation requires actual quantum gates (active operation)

6.1.7 Realization Using Quantum Gates

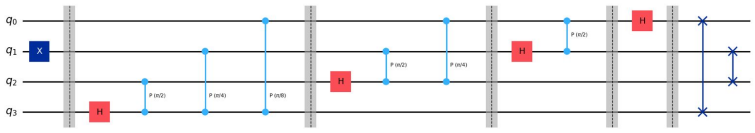

Figure 6.1: 4-qubit Quantum Fourier Transform circuit