Quantum Algorithms

Personal notes from MITx 8.370.2x

Himanshu Dongre

Based on lectures by Peter Shor

These are personal study notes written while following MITx 8.370.2x, taught by Peter Shor. They are not official MIT course materials and are not endorsed by MIT. Any errors are my own.

Contents

General strategy of a quantum algorithm for a classical problem:

1. Take the classical input to the problem.
2. Express the input as a quantum state that is a superposition of an *exponential* number of classical states.
3. Convert it into a quantum state that encodes the solution.
4. Measure to get the solution.

1 Deutsch-Jozsa Algorithm

Deutsch-Jozsa problem: Given a function f on n bits (2^n values), decide whether:

a) f is constant; i.e., $f(x) = 0$ for all xs or $f(x) = 1$ for all xs, or
b) f is balanced; i.e., $f(x) = 0$ for 2^{n-1} xs and $f(x) = 1$ for 2^{n-1} xs.

Note that this is a *promise problem*; i.e., either (a) or (b) is certainly true. How are we given f? We're given f as an *oracle*.

$$x \in \{0,1\}^n \longrightarrow \boxed{f} \longrightarrow f(x) \in \{0,1\}. \tag{1}$$

This is a classical oracle. This cannot possibly be a quantum oracle because it has n bits as input and a single bit as output which makes it irreversible, and a quantum oracle better be reversible. The idea to make it reversible is to keep the input around.

$$|x\rangle|b\rangle \longrightarrow \boxed{\mathcal{O}_{\text{XOR}}} \longrightarrow |x\rangle|b \oplus f(x)\rangle. \tag{2}$$

This is the quantum "XOR" oracle that takes as input string $|x\rangle$ and a bit $|b\rangle$ and returns $|x\rangle$ and $|b \oplus f(x)\rangle$.

1.1 2-bit Quantum Algorithm

For the two bit case, we have four evaluations: $f(00)$, $f(01)$, $f(10)$, $f(11)$. We define a different quantum oracle called the "phase" oracle:

$$|x\rangle \longrightarrow \boxed{\mathcal{O}_{\text{phase}}} \longrightarrow (-1)^{f(x)}|x\rangle. \tag{3}$$

Give as input to the phase oracle the state

$$|{+}{+}\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle). \tag{4}$$

Then, if f is constant, the phase oracle outputs

$$\pm|{+}{+}\rangle = \pm\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle). \tag{5}$$

Otherwise, if f is balanced, then the phase oracle outputs a state with two internal/local phase flips; for instance,

$$\frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle). \tag{6}$$

Hence, the state (5) is the output in case (a) and something like state (6) is the output in case (b). The two cases can thus be distinguished by making the following measurement:

$$|{+}{+}\rangle \longrightarrow \boxed{\mathcal{O}_{\text{phase}}} \longrightarrow \boxed{H \otimes H} \longrightarrow \boxed{\text{Measure}} \longrightarrow b_1 b_2 \in \{0,1\}^2. \tag{7}$$

If measurement output is 00, f is constant. Else, f is balanced.

1.1.1 Analysis

Applying a two-qubit Hadamard gate $H \otimes H$ and measuring in the computational basis is the same as measuring in the following basis:

$$|{+}{+}\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle), \tag{8}$$

$$|{+}{-}\rangle = \frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle), \tag{9}$$

$$|{-}{+}\rangle = \frac{1}{2}(|00\rangle + |01\rangle - |10\rangle - |11\rangle), \tag{10}$$

$$|{-}{-}\rangle = \frac{1}{2}(|00\rangle - |01\rangle - |10\rangle + |11\rangle). \tag{11}$$

If f is constant, then the output of the phase oracle is $|{+}{+}\rangle$, else it's one of the other three up to global phase.

1.2 n-bit Quantum Algorithm

We carry out the following measurement:

$$|+\rangle^{\otimes n} \longrightarrow \boxed{\mathcal{O}_{\text{phase}}} \longrightarrow \boxed{H^{\otimes n}} \longrightarrow \boxed{\text{Measure}} \longrightarrow b_1 b_2 \ldots b_n \in \{0,1\}^n. \tag{12}$$

If f is constant, the measurement result is 0^n. Otherwise, if f is balanced, the measurement result is not 0^n.

1.2.1 Analysis

Our goal here is to find the probability of getting 0^n on measurement. We know that

$$\mathbb{P}(0^n) = \left|\langle +|^{\otimes n} \mathcal{O}_{\text{phase}} |+\rangle^{\otimes n}\right|^2. \tag{13}$$

Also,

$$|+\rangle^{\otimes n} = 2^{-\frac{n}{2}} \sum_{x \in \{0,1\}^n} |x\rangle. \tag{14}$$

Hence,

$$\mathcal{O}_{\text{phase}}|+\rangle^{\otimes n} = 2^{-\frac{n}{2}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}|x\rangle. \tag{15}$$

Therefore, from (13), (14), and (15),

$$\mathbb{P}(0^n) = \left| \frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \right|^2 \tag{16}$$

$$= \frac{1}{2^{2n}} \left(\sum_{x\in\{0,1\}^n} (-1)^{f(x)} \right)^2 \tag{17}$$

$$= \frac{1}{2^{2n}} \begin{cases} (\pm 2^n)^2 & \text{if } f \text{ constant,} \\ 0 & \text{if } f \text{ balanced} \end{cases} \tag{18}$$

$$= \begin{cases} 1 & \text{if } f \text{ constant,} \\ 0 & \text{if } f \text{ balanced.} \end{cases} \tag{19}$$

1.3 Classical Algorithm

We query the classical oracle for inputs $f(a)$, $f(b)$, $f(c)$, ... We need $2^{n-1}+1$ queries to know that f is not constant; and since the DJP is a promise problem, this means that f is balanced. In contrast, quantumly, we only need one query to the phase oracle.

But, historically, this convinced nobody that quantum computers are better...

1.3.1 Drawback

Randomized classical algorithm: Ask random queries until we get $f(i) = 0$ and $f(j) = 1$. Call the output of the first query a "failure". Then, the expected number of additional queries is the expected number of tosses of a fair coin until a "success." This expectation is 2. Hence, we need a total of 3 queries in expectation.

1.4 Runtime Complexity

With an n-bit function f, the number of queries is:

Deterministic classical	Randomized classical	Quantum Deutsch-Jozsa
$2^{n-1}+1 = \Theta(2^n)$	$3 = \Theta(1)$	$1 = \Theta(1)$

1.5 Aside: Phase Oracle versus XOR Oracle

Claim: We can use the XOR oracle to get the phase oracle.

Idea: Send $|b\rangle = |-\rangle$ in the XOR oracle.

$$|x\rangle|-\rangle \longrightarrow \boxed{\mathcal{O}_{\text{XOR}}} \longrightarrow |x\rangle|- \oplus f(x)\rangle \tag{20}$$

$$= |x\rangle \frac{1}{\sqrt{2}}(|f(x)\rangle + |1 \oplus f(x)\rangle) \tag{21}$$

$$= |x\rangle \begin{cases} \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) & \text{if } f(x) = 0, \\ \frac{1}{\sqrt{2}}(|1\rangle - |0\rangle) & \text{if } f(x) = 1 \end{cases} \tag{22}$$

$$= |x\rangle \begin{cases} |-\rangle & \text{if } f(x) = 0, \\ -|-\rangle & \text{if } f(x) = 1 \end{cases} \tag{23}$$

$$= \underbrace{(-1)^{f(x)}|x\rangle}_{\mathcal{O}_{\text{phase}}|x\rangle}|-\rangle. \tag{24}$$

This is called *phase kickback*. We can also get the XOR oracle from the phase oracle, but that construction is more complicated.

2 Simon's Algorithm

Simon's problem: Given a function f (as a black box) such that f is either 1-to-1 or 2-to-1 with $f(x) = f(x \oplus c)$ for all x. Decide which and find c.

Note that Simon's problem is also a promise problem like the DJP. Also, if $c = 0^n$, then f must be 1-to-1, otherwise it's 2-to-1. That is, the problem is to find c if a *nonzero* c exists such that $f(x) = f(x \oplus c)$ for all x, otherwise to conclude that f is 1-to-1. For example, in

x	$f(x)$
00	01
01	11
10	11
11	01

observe that f is 2-to-1 with $c = 11$.

2.1 Classical Algorithm

Much like the DJP, the classical algorithm here is to keep querying the f-oracle until we get some conclusive information. The choice of first query doesn't make a difference because everything is symmetric. Then, we keep querying without getting anywhere until we find a collision; i.e., two inputs whose outputs match. Suppose we get that $f(a) = f(b)$, then $c = a \oplus b$, and f is 2-to-1. The other possibility is that we exhaust all inputs without finding matching outputs. Then, f is 1-to-1.

2.1.1 Runtime Complexity

If we have n-bit inputs, then there are 2^n possible cs. After k queries, we have tried at most as many values of c as the number of 2-combinations of k inputs since each pair of inputs

corresponds to one unique value of c if queried optimally. Hence, after k queries, we have found $\binom{k}{2} = \frac{k(k-1)}{2}$ cs. Note that on the $(k+1)^{\text{th}}$ query, we try exactly k new cs. Hence, if an adversary chooses f so that all cs are equally likely, then the probability of "success" on the $(k+1)^{\text{th}}$ query is $\frac{k}{2^n}$. Let K be the random variable denoting the index of the successful query. Then,

$$\mathbb{P}(K=k) = \underbrace{\left[\prod_{m=1}^{k-1}\left(1-\frac{m-1}{2^n}\right)\right]}_{\text{failures}}\underbrace{\left(\frac{k-1}{2^n}\right)}_{\text{success}}. \tag{25}$$

Hence, the expectation $\mathbb{E}[K] = \sum_{k=1}^{2^n} k\mathbb{P}(K=k)$ can be shown to be $\Theta\left(2^{\frac{n}{2}}\right)$. It is analogous to how a collision in a hash table with N entries can be found in $\Theta\left(\sqrt{N}\right)$ time.

Shor's analysis: The value of c is one out of a list of 2^n possible values. If we try the values at random, then we would expect to arrive at the right value about halfway through the list. Hence, if we run k queries such that 2^{n-1} values of c are tried, then we would expect a success within them. Approximately $\frac{k^2}{2}$ values of c are tried in k queries. Hence,

$$\frac{k^2}{2} = 2^{n-1} \Rightarrow k = 2^{\frac{n}{2}}. \tag{26}$$

Modern analysis: After k queries, we have queried k inputs and thus formed about $\binom{k}{2} \approx \frac{k^2}{2}$ queries pairs. A pair (x, y) reveals the hidden mask only if $y = x \oplus c$, and for a randomly chosen pair this happens with probability about 2^{-n}. Hence, the expected number of useful collisions is $\frac{k^2}{2} \cdot \frac{1}{2^n} \approx \frac{k^2}{2^n}$. Setting this equal to 1 gives $k = \Theta(2^{\frac{n}{2}})$.

Therefore, any classical algorithm takes $\Theta\left(2^{\frac{n}{2}}\right)$ queries.

2.2 Aside: Result of $H^{\otimes n}$

Suppose x is a bit string of length n. Then,

$$H^{\otimes n}|x\rangle = \bigotimes_{i=1}^{n} \begin{cases} |+\rangle & \text{if } x^{(i)} = 0, \\ |-\rangle & \text{if } x^{(i)} = 1. \end{cases} \tag{27}$$

Hence,

$$H^{\otimes n}|0\rangle^{\otimes n} = |+\rangle^{\otimes n} = 2^{-\frac{n}{2}} \sum_{y\in\{0,1\}^n} |y\rangle. \qquad \text{from (14)} \tag{28}$$

If the i^{th} bit of x is 1, then the i^{th} qubit of $H^{\otimes n}|x\rangle$ is in state $|-\rangle$. How does this change the summation in the RHS of equation (28)? The state of the i^{th} qubit of $H^{\otimes n}|x\rangle$ ($|+\rangle$ or $|-\rangle$) contributes the i^{th} bit of every bit string y in the summand. Hence, the only difference is in the way $|+\rangle$ and $|-\rangle$ contribute a 1: $|-\rangle$ does it with a factor of -1. Thus, if the i^{th} qubit of $H^{\otimes n}|x\rangle$ is in state $|-\rangle$, and the i^{th} bit of y is 1 (i.e., a 1 has been contributed), then $|y\rangle$

picks up a factor of -1. Equivalently, if $x^{(i)} = 1$ and $y^{(i)} = 1$, then $|y\rangle$ picks up a factor of -1. Therefore,

$$H^{\otimes n}|x\rangle = 2^{-\frac{n}{2}} \sum_{y\in\{0,1\}^n} (-1)^{\sum_{i=1}^n x^{(i)}y^{(i)}}|y\rangle \tag{29}$$

$$= 2^{-\frac{n}{2}} \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}|y\rangle. \tag{30}$$

2.3 Quantum Algorithm

Step 1: Start with the state

$$|+\rangle^{\otimes n}|0\rangle^{\otimes n} = 2^{-\frac{n}{2}} \sum_{x\in\{0,1\}^n} |x\rangle|0^n\rangle, \tag{31}$$

where we have n ancilla qubits set to $|0\rangle$s.

Step 2: We assume access to an oracle that lets us compute f quantumly as follows:

$$|x\rangle|y\rangle \longrightarrow \boxed{\mathcal{O}_f} \longrightarrow |x\rangle|y \oplus f(x)\rangle. \tag{32}$$

Query the oracle with state $|+\rangle^{\otimes n}|0\rangle^{\otimes n}$:

$$\mathcal{O}_f|+\rangle^{\otimes n}|0\rangle^{\otimes n} = 2^{-\frac{n}{2}} \sum_{x\in\{0,1\}^n} |x\rangle|f(x)\rangle. \tag{33}$$

Step 3: Take $H^{\otimes n}$ on the first n qubits:

$$\left(H^{\otimes n} \otimes I^{\otimes n}\right)\mathcal{O}_f|+\rangle^{\otimes n}|0\rangle^{\otimes n} = 2^{-\frac{n}{2}} \sum_{x\in\{0,1\}^n} H^{\otimes n}|x\rangle|f(x)\rangle \tag{34}$$

$$= 2^{-n} \sum_{x\in\{0,1\}^n} \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}|y\rangle|f(x)\rangle. \tag{35}$$

Step 4: Measure the first n qubits to get a value of y. This measurement does not give the answer, but gives information about the answer.

Repeat steps 1 through 4 $\Theta(n)$ times and compute c classically from all the measurement outcomes obtained.

2.3.1 Analysis

What information does y give us about f? To answer this question, we must first ask what is the probability of seeing a particular y_0 as a measurement outcome? To answer this, group the terms in the double sum in (35) which have $|y_0\rangle$, then look at the coefficient on it and square it. Since $|y_0\rangle$ does not depend on x, the grouping of terms we're looking for is

$$2^{-n}|y_0\rangle \sum_{x\in\{0,1\}^n} (-1)^{x\cdot y_0}|f(x)\rangle. \tag{36}$$

Thus,

$$\mathbb{P}(y_0) = \left\| 2^{-n} \sum_{x\in\{0,1\}^n} (-1)^{x\cdot y_0} |f(x)\rangle \right\|^2 \tag{37}$$

$$= 4^{-n} \left\| \sum_{\text{pairs } \{x,x\oplus c\}} \left[(-1)^{x\cdot y_0} + (-1)^{(x\oplus c)\cdot y_0} \right] |f(x)\rangle \right\|^2 \tag{38}$$

$$= 4^{-n} \left\| \sum_{\text{pairs } \{x,x\oplus c\}} \left[(-1)^{x\cdot y_0} + (-1)^{x\cdot y_0 \oplus c\cdot y_0} \right] |f(x)\rangle \right\|^2 \tag{39}$$

$$= 4^{-n} \left\| \sum_{\text{pairs } \{x,x\oplus c\}} (-1)^{x\cdot y_0} [1 + (-1)^{c\cdot y_0}] |f(x)\rangle \right\|^2 . \tag{40}$$

In the 2-to-1 case, if $c \cdot y_0 = 1$, then the coefficient of $|f(x)\rangle$ vanishes, and, as a result, the sum vanishes. Hence,

$$\mathbb{P}(y_0) = 0 \quad \text{if } c \cdot y_0 = 1. \tag{41}$$

If $c \cdot y_0 = 0$, then

$$\mathbb{P}(y_0) = 4^{-n} \left\| \sum_{\text{pairs } \{x,x\oplus c\}} 2(-1)^{x\cdot y_0} |f(x)\rangle \right\|^2 \tag{42}$$

$$= 4^{-n+1} \left\| \sum_{\text{pairs } \{x,x\oplus c\}} (-1)^{x\cdot y_0} |f(x)\rangle \right\|^2 \tag{43}$$

$$= 4^{-n+1} \left(\sum_{\text{pairs } \{x',x'\oplus c\}} (-1)^{x'\cdot y_0} \langle f(x')| \right) \sum_{\text{pairs } \{x,x\oplus c\}} (-1)^{x\cdot y_0} |f(x)\rangle \tag{44}$$

$$= 4^{-n+1} \sum_{\text{pairs } \{x',x'\oplus c\}} \sum_{\text{pairs } \{x,x\oplus c\}} (-1)^{x'\cdot y_0 \oplus x\cdot y_0} \langle f(x')|f(x)\rangle \tag{45}$$

$$= 4^{-n+1} \sum_{\text{pairs } \{x',x'\oplus c\}} \sum_{\text{pairs } \{x,x\oplus c\}} (-1)^{x'\cdot y_0 \oplus x\cdot y_0} \delta_{xx'} \tag{46}$$

$$= 4^{-n+1} \sum_{\text{pairs } \{x,x\oplus c\}} (-1)^{2x\cdot y_0} \tag{47}$$

$$= 4^{-n+1} \sum_{\text{pairs } \{x,x\oplus c\}} 1 \tag{48}$$

$$= 4^{-n+1} 2^{n-1} \tag{49}$$

$$= 2^{-n}. \tag{50}$$

So, from (41) and (50), we only get ys in the measurement outcomes that are perpendicular to c. Hence, each y gives one linear equation over $\mathbb{F}_2$ that c must satisfy. We need enough samples $y_1, \ldots, y_m$ to obtain $n-1$ independent equations $c \cdot y_i = 0$. Then, c is the unique nonzero vector orthogonal to all of them. So the classical postprocessing step is to collect

these equations and solve a linear system over $\mathbb{F}_2$. Therefore, we need $\Theta(n)$ such iterations to get c.

What is the probability that we find c in the first $n-1$ iterations? This happens only if all of the first $n-1$ y_is turn out to be linearly independent. The outcomes are uniform over the 2^{n-1} bit strings perpendicular to c. Thus, y_1 must be anything but 0^n, the probability of which is $1-2^{-(n-1)}$. After k independent outcomes have been obtained, they span a k-dimensional subspace containing exactly 2^k vectors, so the next outcome is independent with probability $1-2^{k-(n-1)}$. Therefore,

$$\mathbb{P}(c \text{ found in } n-1 \text{ iterations}) = \prod_{k=0}^{n-2}(1-2^{k-(n-1)}) \tag{51}$$

$$= \left(1-\frac{1}{2}\right)\left(1-\frac{1}{4}\right)\ldots\left(1-\frac{1}{2^{n-1}}\right). \tag{52}$$

For large n, this is approximately

$$\prod_{j=1}^{\infty}(1-2^{-j}) \approx 0.2888. \tag{53}$$

The expected number of iterations required to find c is $n+O(1)$.

Note: In the case that f is 1-to-1, the c obtained this way would be incorrect because we completely discount the case that $c=0^n$. Hence, one additional check is needed at the end to conclude whether f is 2-to-1 with the c obtained, or 1-to-1. We check whether $f(0) \stackrel{?}{=} f(c)$.

3 Classical Fourier Transform On A Finite Abelian Group

Let G be a finite Abelian group, written additively. The goal of this section is to define the Fourier transform on G from first principles. We begin with functions on G, introduce characters, derive their general form using the fundamental theorem of finite Abelian groups, and then define the Fourier transform.

3.1 Function Space

Let

$$\mathbb{C}[G] = \{f : G \to \mathbb{C}\} \tag{54}$$

denote that vector space of all complex-valued functions on G. Since G is finite, each function $f \in \mathbb{C}[G]$ is determined by its values on the elements of G. If $G = \left\{g_1, \ldots, g_{|G|}\right\}$, then specifying f is equivalent to specifying the $|G|$-tuple

$$(f(g_1), \ldots, f(g_{|G|})) \in \mathbb{C}^{|G|}. \tag{55}$$

Thus, $\mathbb{C}[G]$ is naturally isomorphic, as a complex vector space, to $\mathbb{C}^{|G|}$. In particular,

$$\dim \mathbb{C}[G] = |G|. \tag{56}$$

For each $g \in G$, define the delta function

$$\delta_g(h) = \begin{cases} 1 & \text{if } h = g, \\ 0 & \text{if } h \neq g. \end{cases} \tag{57}$$

Then the set $\{\delta_g \mid g \in G\}$ is a basis of $\mathbb{C}[G]$. So the function space on G has one complex degree of freedom per group element.

3.2 Characters

Definition. A *character of* G is a group homomorphism

$$\chi : G \to \mathbb{C}^\times \tag{58}$$

where $\mathbb{C}^\times = \mathbb{C} \setminus \{0\}$ is the multiplicative group of nonzero complex numbers. Thus, χ satisfies

$$\chi(g + h) = \chi(g)\chi(h) \qquad \text{for all } g, h \in G \tag{59}$$

The characters of G form a group under pointwise multiplication:

$$(\chi_1\chi_2)(g) = \chi_1(g)\chi_2(g). \tag{60}$$

This group is denoted by $\widehat{G}$ and is called the *dual group of* G, or in this context the *Pontryagin dual of* G. Although Pontryagin duality is usually discussed for locally compact Abelian (LCA) groups, the finite Abelian case is the simplest and most concrete instance of it.

3.2.1 Properties of characters

Lemma. If $\chi \in \widehat{G}$, then for every $g \in G$,

$$\chi(0) = 1, \quad \chi(-g) = \chi(g)^{-1}, \tag{61}$$

and $\chi(g)$ is a root of unity.

Proof. Since χ is a group homomorphism,

$$\chi(0) = \chi(0 + 0) = \chi(0)\chi(0), \tag{62}$$

so $\chi(0) = 1$. Next,

$$1 = \chi(0) = \chi(g + (-g)) = \chi(g)\chi(-g), \tag{63}$$

hence $\chi(-g) = \chi(g)^{-1}$. Finally, since G is finite, every $g \in G$ has finite order, so $|g|g = 0$. Therefore, if $m = |g|$, then

$$\chi(g)^m = \chi(mg) = \chi(0) = 1. \tag{64}$$

Thus, $\chi(g)$ is an m^{th} root of unity.□

So characters are phase-valued functions that convert addition in the group into multiplication of phases.

3.2.2 Example: Characters of a cyclic group

We begin with the basic case $G = \mathbb{Z}_n$.

Proposition. Every character of $\mathbb{Z}_n$ has the form

$$\chi_k(j) = \exp\left(\frac{2\pi\iota kj}{n}\right), \quad k \in \mathbb{Z}_n. \tag{65}$$

Proof. A character $\chi : \mathbb{Z}_n \to \mathbb{C}^\times$ is determined entirely by the value $\chi(1)$, since every $j \in \mathbb{Z}_n$ can be written as $j \cdot 1$. By the homomorphism property, $\chi(j) = \chi(1)^j$. Now, $n \cdot 1 = 0$ in $\mathbb{Z}_n$, so

$$1 = \chi(0) = \chi(n \cdot 1) = \chi(1)^n. \tag{66}$$

Thus, $\chi(1)$ is an n^{th} root of unity, so

$$\chi(1) = \exp\left(\frac{2\pi\iota k}{n}\right) \tag{67}$$

for some $k \in \{0, 1, \ldots, n-1\}$. Therefore,

$$\chi(j) = \exp\left(\frac{2\pi\iota kj}{n}\right). \tag{68}$$

Conversely, each such formula defines a homomorphism, since

$$\chi_k(j+\ell) = \exp\left(\frac{2\pi\iota k(j+\ell)}{n}\right) = \exp\left(\frac{2\pi\iota kj}{n}\right)\exp\left(\frac{2\pi\iota k\ell}{n}\right) = \chi_k(j)\chi_k(\ell). \qquad \square \tag{69}$$

3.2.3 Characters of a direct sum

We now pass to a general finite Abelian group. By the fundamental theorem for finite Abelian groups, there exist positive integers $n_1, \ldots, n_r$ such that

$$G \cong \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_r}. \tag{70}$$

Thus, every element $g \in G$ may be written in coordinates as

$$g = (g_1, \ldots, g_r), \quad g_i \in \mathbb{Z}_{n_i}. \tag{71}$$

We derive the general formula for a character on such a group.

Theorem. Let $G = \bigoplus_{i=1}^r \mathbb{Z}_{n_i}$. Then, every character $\chi \in \widehat{G}$ is of the form

$$\chi_k(g_1, \ldots, g_r) = \exp\left(2\pi\iota \sum_{i=1}^r \frac{k_i g_i}{n_i}\right), \quad k = (k_1, \ldots, k_r) \in G. \tag{72}$$

Proof. Let $e_1, \ldots, e_r$ denote the standard generators

$$e_1 = (1, 0, \ldots, 0) \quad \ldots, \quad e_r = (0, \ldots, 0, 1). \tag{73}$$

Every element $g = (g_1, \ldots, g_r)$ may be written as

$$g = g_1 e_1 + \cdots + g_r e_r. \tag{74}$$

Since χ is a homomorphism,

$$\chi(g) = \chi\left(\sum_{i=1}^{r} g_r e_i\right) = \prod_{i=1}^{r} \chi(e_i)^{g_i}. \tag{75}$$

Now each e_i has order n_i, so $n_i e_i = 0$. Applying χ,

$$\chi(e_i)^{n_i} = \chi(0) = 1. \tag{76}$$

Hence, $\chi(e_i)$ is an n_i^{th} root of unity, so there exists $k_i \in \{0, 1, \ldots, n_i - 1\}$ such that

$$\chi(e_i) = \exp\left(\frac{2\pi\iota k_i}{n_i}\right). \tag{77}$$

Substituting in (75),

$$\chi(g_1, \ldots, g_r) = \prod_{i=1}^{r} \exp\left(\frac{2\pi\iota k_i}{n_i}\right)^{g_i} = \exp\left(2\pi\iota \sum_{i=1}^{r} \frac{k_i g_i}{n_i}\right). \tag{78}$$

Conversely, any function of this form is a character because

$$\chi_k(g + h) = \exp\left(2\pi\iota \sum_{i=1}^{r} \frac{k_i(g_i + h_i)}{n_i}\right) = \chi_k(g)\chi_k(h). \qquad \square \tag{79}$$

This is the general character formula for a finite Abelian group once characters have been chosen. The formula above shows that characters are indexed by

$$k = (k_1, \ldots, k_r) \in \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_r}. \tag{80}$$

Therefore,

$$\left|\widehat{G}\right| = n_1 \cdots n_r = |G|. \tag{81}$$

So the Pontryagin dual $\widehat{G}$ has the same number of elements as G. Since $\dim \mathbb{C}[G] = |G|$, the number of characters matches the dimension of the function space. This is exactly what one would hope for: there are enough characters to serve as a basis for $\mathbb{C}[G]$. In fact, for finite Abelian groups, one has a noncanonical isomorphism

$$\widehat{G} \cong G \tag{82}$$

though this isomorphism depends on the chosen decomposition into cyclic factors.

3.2.4 Orthogonality of characters

Proposition. For characters $\chi, \eta \in \widehat{G}$,

$$\sum_{g \in G} \chi(g)\eta(g)^* = \begin{cases} |G| & \text{if } \chi = \eta, \\ 0 & \text{if } \chi \neq \eta. \end{cases} \tag{83}$$

Proof. If $\chi = \eta$, then

$$\chi(g)\eta(g)^* = \chi(g)\chi(g)^* = |\chi(g)|^2 = 1 \tag{84}$$

for every $g \in G$, so the sum is $|G|$. Now assume $\chi \neq \eta$. Then, $\xi = \chi\eta^*$ is a nontrivial character. Consider

$$S = \sum_{g \in G} \xi(g). \tag{85}$$

Since ξ is nontrivial, there exists $h \in G$ such that $\xi(h) \neq 1$. Then,

$$\xi(h)S = \sum_{g \in G} \xi(h)\xi(g) = \sum_{g \in G} \xi(h+g). \tag{86}$$

As g runs over G, so does $h + g$, hence

$$\xi(h)S = S. \tag{87}$$

Therefore,

$$(\xi(h) - 1)S = 0. \tag{88}$$

Since $\xi(h) \neq 1$, it follows that $S = 0$. Thus,

$$\sum_{g \in G} \chi(g)\eta(g)^* = 0. \qquad \square \tag{89}$$

So the normalized characters

$$\frac{1}{\sqrt{|G|}}\chi \qquad (\chi \in \widehat{G}) \tag{90}$$

form an orthonormal set in $\mathbb{C}[G]$. Since there are $\left|\widehat{G}\right| = \dim \mathbb{C}[G]$ of them, they form an orthonormal basis.

3.3 Fourier Transform

We can now define the Fourier transform on G.

Definition. Let G be a finite Abelian group and let $\widehat{G}$ be its Pontryagin dual. For $f \in \mathbb{C}[G]$, its *Fourier transform* is the function $\widehat{f} : \widehat{G} \to \mathbb{C}$ defined by

$$\widehat{f}(\chi) = \frac{1}{\sqrt{|G|}} \sum_{g \in G} \chi(g)^* f(g). \tag{91}$$

Equivalently, if we identify $G \cong \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_r}$ and index the characters by $k = (k_1, \ldots, k_r)$, then

$$\widehat{f}(\chi) = \frac{1}{\sqrt{|G|}} \sum_{g \in G} \exp\left(-2\pi\iota \sum_{i=1}^{r} \frac{k_i g_i}{n_i}\right) f(g), \tag{92}$$

where $g = (g_1, \ldots, g_r)$.

This is the general discrete Fourier transform on a finite Abelian group. It's clear from the formula that a character is the analogue of the exponential function from the standard discrete Fourier transform, and the reason characters are the correct Fourier basis is their orthogonality. Because the normalized characters form an orthonormal basis, every function $f \in \mathbb{C}[G]$ admits an expansion in characters.

Theorem (*Inversion formula*). For every $f \in \mathbb{C}[G]$,

$$f(g) = \frac{1}{\sqrt{|G|}} \sum_{\chi \in \widehat{G}} \chi(g) \widehat{f}(\chi). \tag{93}$$

Proof. Since the normalized characters form an orthonormal basis of $\mathbb{C}[G]$, this is simply the standard expansion of a vector in an orthonormal basis. Explicitly, if

$$\tilde{\chi} = \frac{1}{\sqrt{|G|}} \chi, \tag{94}$$

then

$$f = \sum_{\chi \in \widehat{G}} \langle \tilde{\chi}, f \rangle \tilde{\chi}. \tag{95}$$

Now,

$$\langle \tilde{\chi}, f \rangle = \sum_{g \in G} \tilde{\chi}(g)^* f(g) = \frac{1}{\sqrt{|G|}} \sum_{g \in G} \chi(g)^* f(g) = \widehat{f}(\chi). \tag{96}$$

Therefore,

$$f(g) = \frac{1}{\sqrt{|G|}} \sum_{\chi \in \widehat{G}} \widehat{f}(\chi) \chi(g). \qquad \square \tag{97}$$

3.3.1 Example 1: FT on a cyclic group

Here

$$G = \mathbb{Z}_n, \quad f \in \mathbb{C}[\mathbb{Z}_n], \quad \chi_k(j) = \exp\left(\frac{2\pi\iota k j}{n}\right). \tag{98}$$

Thus,

$$\widehat{f}(k) = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} \exp\left(-\frac{2\pi\iota k j}{n}\right) f(j), \tag{99}$$

which is the usual discrete Fourier transform (DFT).

3.3.2 Example 2: FT on the group of bit strings

Here

$$G = \mathbb{Z}_2^n, \quad f \in \mathbb{C}[\mathbb{Z}_2^n]. \tag{100}$$

Writing $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$, the general character formula gives

$$\chi_y(x) = \exp\left(2\pi\iota \sum_{i=1}^{n} \frac{y_i x_i}{2}\right) = \exp\left(\pi\iota \sum_{i=1}^{n} y_i x_i\right) = e^{\pi\iota(x\cdot y)} = (-1)^{x\cdot y}. \tag{101}$$

Hence,

$$\widehat{f}(\chi_y) = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{x\cdot y} f(x). \tag{102}$$

3.4 Summary

For a finite Abelian group G, the Fourier transform is built from its characters, i.e., from the elements of its Pontryagin dual

$$\widehat{G} = \mathsf{Hom}(G, \mathbb{C}^\times). \tag{103}$$

The key steps are:

1. $\mathbb{C}[G]$ is the $|G|$-dimensional vector space of all complex-valued functions on G.
2. A character is a homomorphism $G \to \mathbb{C}^\times$.
3. If $G \cong \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_r}$, then every character is of the form

$$\chi_k(g) = \exp\left(2\pi\iota \sum_{i=1}^{r} \frac{g_i k_i}{n_i}\right). \tag{104}$$

4. The characters are orthogonal and there are exactly $|G|$ of them.
5. Therefore they form an orthonormal basis of $\mathbb{C}[G]$, and the Fourier transform is the expansion of a function in that basis:

$$\widehat{f}(\chi) = \frac{1}{\sqrt{|G|}} \sum_{g\in G} \chi(g)^* f(g). \tag{105}$$

One important subtlety: the explicit coordinate formula for χ_k depends on a chosen decomposition

$$G \cong \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_r}. \tag{106}$$

So the formula is not canonical in coordinates, even though the dual group $\widehat{G}$ and the Fourier transform itself are canonical.

4 Quantum Fourier Transform

Let G be a finite Abelian group, written additively. The classical Fourier transform on G takes a function $f : G \to \mathbb{C}$ and expands it in the character basis. In quantum computing, we package the same construction into a unitary operator acting on a Hilbert space with computational basis indexed by G.

4.1 Hilbert Space

Let

$$\mathcal{H}_G = \mathsf{span}\{|g\rangle \mid g \in G\} \tag{107}$$

be the complex Hilbert space with orthonormal basis labeled by the elements of G. Since there is one basis vector for each group element,

$$\dim \mathcal{H}_G = |G|. \tag{108}$$

A general state in $\mathcal{H}_G$ has the form

$$|\psi\rangle = \sum_{g\in G} \alpha_g |g\rangle, \qquad \alpha_g \in \mathbb{C}, \tag{109}$$

with normalization

$$\sum_{g\in G} |\alpha_g|^2 = 1. \tag{110}$$

4.2 Characters as Quantum Phase Functions

Recall that the Pontryagin dual of G is

$$\widehat{G} = \mathsf{Hom}(G, \mathbb{C}^\times), \tag{111}$$

the group of characters of G. Each character $\chi \in \widehat{G}$ is a function $\chi : G \to \mathbb{C}^\times$ satisfying

$$\chi(g + h) = \chi(g)\chi(h). \tag{112}$$

Since G is finite, each $\chi(g)$ is a root of unity, hence a complex phase. Thus characters are exactly the functions that assign compatible phases to group elements. From the classical point of view, the characters form an orthogonal basis of $\mathbb{C}[G]$. In the quantum setting, they determine a new orthonormal basis of $\mathcal{H}_G$.

4.3 Fourier Basis States

For each character $\chi \in \widehat{G}$, define the state

$$|e_\chi\rangle = \frac{1}{\sqrt{|G|}} \sum_{g\in G} \chi(g)|g\rangle. \tag{113}$$

These are the Fourier basis states.

Proposition. The set $\left\{|e_\chi\rangle \mid \chi \in \widehat{G}\right\}$ is an orthonormal basis of $\mathcal{H}_G$.

Proof. For $\chi, \eta \in \widehat{G}$,

$$\langle e_\chi | e_\eta \rangle = \frac{1}{|G|} \sum_{g,h \in G} \chi(g)^* \eta(h) \langle g|h\rangle \tag{114}$$

$$= \frac{1}{|G|} \sum_{g,h \in G} \chi(g)^* \eta(h) \delta_{gh} \tag{115}$$

$$= \frac{1}{|G|} \sum_{g \in G} \chi(g)^* \eta(g) \tag{116}$$

$$= \frac{1}{|G|} \begin{cases} |G| & \text{if } \chi = \eta, \\ 0 & \text{if } \chi \neq \eta \end{cases} \tag{117}$$

$$= \delta_{\chi\eta}. \tag{118}$$

Since there are $\left|\widehat{G}\right| = |G|$ such states, they form an orthonormal basis of the $|G|$-dimensional space $\mathcal{H}_G$.□

So the computational basis is the group elements rewritten as quantum states, and the Fourier basis is the character basis rewritten as quantum states. Note that the basis $\{e_\chi\}_{\chi \in \widehat{G}}$ is a basis of $\mathcal{H}_G$ indexed by $\widehat{G}$. We denote the standard basis of $\mathcal{H}_{\widehat{G}}$ by $\{|\chi\rangle\}_{\chi \in \widehat{G}}$.

4.4 Definition of the Quantum Fourier Transform

The quantum Fourier transform (QFT) is the unitary that maps the computational basis to the Fourier basis.

Definition. The *quantum Fourier transform* over G is the linear operator

$$\mathcal{F}_G : \mathcal{H}_G \to \mathcal{H}_{\widehat{G}} \tag{119}$$

defined on basis states by

$$\mathcal{F}_G |g\rangle = \frac{1}{\sqrt{|G|}} \sum_{\chi \in \widehat{G}} \chi(g) |\chi\rangle, \tag{120}$$

where $\mathcal{H}_{\widehat{G}}$ is the Hilbert space with orthonormal basis indexed by the characters of G.

Since $\left|\widehat{G}\right| = |G|$, the domain and codomain have the same dimension, so this can be viewed as a square matrix once bases are chosen.

Proposition. The operator $\mathcal{F}_G$ is unitary.

Proof. It suffices to show that the images of the computational basis states are orthonormal. For $g, h \in G$,

$$\langle h | \mathcal{F}_G^\dagger \mathcal{F}_G | g \rangle = \frac{1}{|G|} \sum_{\chi \in \widehat{G}} \chi(h)^* \chi(g) = \frac{1}{|G|} \sum_{\chi \in \widehat{G}} \chi(g - h). \tag{121}$$

Now use the dual orthogonality relation:

$$\sum_{\chi \in \widehat{G}} \chi(x) = \begin{cases} |G| & \text{if } x = 0 \\ 0 & \text{if } x \neq 0. \end{cases} \tag{122}$$

Therefore,

$$\langle h | \mathcal{F}_G^\dagger \mathcal{F}_G | g \rangle = \delta_{hg}. \tag{123}$$

Hence $\mathcal{F}_G^\dagger \mathcal{F}_G = I$, so $\mathcal{F}_G$ is unitary.$\square$

So the quantum Fourier transform is not some new object separate from the classical Fourier transform. It is the same change of basis, now interpreted as a unitary map between Hilbert spaces.

4.4.1 Coordinate form

Using the fundamental theorem of finite Abelian groups, suppose

$$G \cong \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_r}. \tag{124}$$

Then every group element may be written as

$$g = (g_1, \ldots, g_r), \quad g_i \in \mathbb{Z}_{n_i}, \tag{125}$$

and every character is indexed by

$$k = (k_1, \ldots, k_r) \in G \tag{126}$$

with

$$\chi_k(g) = \exp\left(2\pi\iota \sum_{i=1}^{r} \frac{g_i k_i}{n_i} \right). \tag{127}$$

Substituting this into the abstract definition gives

$$\mathcal{F}_G |g_1, \ldots, g_r\rangle = \frac{1}{\sqrt{|G|}} \sum_{k \in G} \exp\left(2\pi\iota \sum_{i=1}^{r} \frac{g_i k_i}{n_i} \right) |k_1, \ldots, k_r\rangle. \tag{128}$$

This is an explicit coordinate formula for the QFT over a finite Abelian group.

4.5 Example: The Group of Bit Strings

Let $G = \mathbb{Z}_2^n$. An element is a bit string

$$x = (x_1, \ldots, x_n), \quad x_i \in \{0, 1\}, \tag{129}$$

with group operation given by bitwise addition mod 2, i.e., XOR. The characters are indexed by binary strings $y = (y_1, \ldots, y_n)$ and have the form

$$\chi_y(x) = \exp\left(2\pi\iota \sum_{i=1}^{n} \frac{x_i y_i}{2}\right) = \exp\left(\pi\iota \sum_{i=1}^{n} x_i y_i\right) = e^{\pi\iota(x\cdot y)} = (-1)^{x\cdot y}. \tag{130}$$

Thus,

$$\mathcal{F}_{\mathbb{Z}_2^n}|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}|y\rangle. \tag{131}$$

But this is exactly the action of $H^{\otimes n}$. Therefore,

$$\mathcal{F}_{\mathbb{Z}_2^n} = H^{\otimes n}. \tag{132}$$

So the n-qubit Hadamard transform is precisely the QFT over the additive group of binary strings.

4.6 Interpretation

The formula

$$\mathcal{F}_G|g\rangle = \frac{1}{\sqrt{|G|}} \sum_{\chi\in\widehat{G}} \chi(g)|\chi\rangle \tag{133}$$

has a very clean meaning:

1. the computational basis $\{|g\rangle \,|\, g \in G\}$ is the basis of group elements;
2. the Fourier basis $\left\{|\chi\rangle \,|\, |\chi\rangle \in \widehat{G}\right\}$ is the basis of frequencies;
3. the matrix entries of the QFT are given by the evaluation of characters:

$$(\mathcal{F}_G)_{kj} = \frac{1}{\sqrt{|G|}}\chi_k(j). \tag{134}$$

In other words, the QFT is a change of basis from the "position basis" indexed by G to the "frequency basis" indexed by its Pontryagin dual $\widehat{G}$.

4.7 Relation to the Classical Fourier Transform

If a state is written as

$$|\psi\rangle = \sum_{g\in G} f(g)|g\rangle, \tag{135}$$

then applying $\mathcal{F}_G$ gives

$$\mathcal{F}_G|\psi\rangle = \frac{1}{\sqrt{|G|}} \sum_{\chi\in\widehat{G}} \left(\sum_{g\in G} f(g)\chi(g)\right)|\chi\rangle. \tag{136}$$

Depending on convention, one often puts a complex conjugate on $\chi(g)$ in the classical Fourier transform. That difference is just a sign convention between forward and inverse transforms. The structural point is unchanged:

> the amplitudes in the Fourier basis are the Fourier coefficients of the function f.

So the classical and quantum Fourier transforms are the same change of basis, interpreted in two different languages.

4.8 Summary

For a finite Abelian group G, the quantum Fourier transform is the unitary

$$\mathcal{F}_G|g\rangle = \frac{1}{\sqrt{|G|}} \sum_{\chi \in \widehat{G}} \chi(g)|\chi\rangle, \tag{137}$$

where $\widehat{G}$ is the Pontryagin dual of G; i.e., the group of characters

$$\widehat{G} = \mathsf{Hom}(G, \mathbb{C}^\times). \tag{138}$$

After choosing a decomposition $G \cong \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_r}$, this becomes

$$\mathcal{F}_G|g_1, \ldots, g_r\rangle = \frac{1}{\sqrt{|G|}} \sum_{k \in G} \exp\left(2\pi\iota \sum_{i=1}^{r} \frac{g_i k_i}{n_i}\right)|k_1, \ldots, k_r\rangle. \tag{139}$$

One central special case is:

$$\mathcal{F}_{\mathbb{Z}_2^n} = H^{\otimes n}. \tag{140}$$

One subtlety: writing χ assumes we have chosen a basis indexed by characters. In practice, one usually identifies $\widehat{G} \cong G$ after choosing coordinates, and then writes the output basis states as $|k\rangle$. That identification is convenient, but not canonical.

Note: Conventionally, if no group is specified, the QFT gate on n qubits is defined to be the QFT on the group $\mathbb{Z}_{2^n}$. More generally, the QFT gate on n qudits is defined to be the QFT on the group $\mathbb{Z}_{d^n}$.

5 Shor's Factoring Algorithm

The quantum Fourier transform is a helpful tool in specifying quantum algorithms. For instance, the $H^{\otimes n}$ transform in Simon's algorithm is a QFT in disguise viz. the QFT on the group of binary strings. Similarly, we need a special case of the QFT in Shor's algorithm as well, viz. the inverse QFT on the cyclic group $\mathbb{Z}_N$. So we begin by explicitly developing that.

5.1 Aside: Inverse QFT on $\mathbb{Z}_N$

The example of the classical Fourier transform on the cyclic group $\mathbb{Z}_N$—commonly called the discrete Fourier transform (DFT)—is covered in the section on the classical Fourier transform. We pick the following fact from that example that also applies in the quantum case: the characters of $\mathbb{Z}_N$ are N^{th} roots of unity:

$$\chi_k(j) = \exp\left(\frac{2\pi\iota kj}{N}\right), \quad k, j \in \mathbb{Z}_N. \tag{141}$$

Hence,

$$\mathcal{F}^\dagger_{\mathbb{Z}_N}|j\rangle = \frac{1}{\sqrt{N}} \sum_{k\in\mathbb{Z}_N} \exp\left(-\frac{2\pi\iota kj}{N}\right)|k\rangle. \tag{142}$$

What does the matrix for this look like? Let $\omega_N = \exp\left(\frac{2\pi\iota}{N}\right)$, and $\zeta_N = \omega_N^* = \exp\left(-\frac{2\pi\iota}{N}\right)$. Then, $\left(\mathcal{F}^\dagger_{\mathbb{Z}_N}\right)_{kj} = \frac{1}{\sqrt{N}}\zeta_N^{kj}$; i.e.,

$$\mathcal{F}^\dagger_{\mathbb{Z}_N} = \frac{1}{\sqrt{N}} \begin{array}{c} \\ k=0 \\ 1 \\ 2 \\ 3 \\ \vdots \\ N-1 \end{array} \begin{array}{c} \begin{array}{cccccc} j=0 & 1 & 2 & 3 & \dots & N-1 \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & \zeta_N & \zeta_N^2 & \zeta_N^3 & \dots & \zeta_N^{N-1} \\ 1 & \zeta_N^2 & \zeta_N^4 & \zeta_N^6 & \dots & \zeta_N^{2(N-1)} \\ 1 & \zeta_N^3 & \zeta_N^6 & \zeta_N^9 & \dots & \zeta_N^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta_N^{N-1} & \zeta_N^{(N-1)2} & \zeta_N^{(N-1)3} & \dots & \zeta_N^{(N-1)^2} \end{bmatrix} \end{array}. \tag{143}$$

Moreover, $\mathcal{F}^\dagger_{\mathbb{Z}_N}$ is unitary, which is why we're writing $\mathcal{F}^{-1}_{\mathbb{Z}_N} = \mathcal{F}^\dagger_{\mathbb{Z}_N}$ in the first place. Hence, there exists a quantum circuit that produces the inverse QFT, i.e. the matrix operator in (143). The question is whether we can come up with an *efficient* circuit, for instance, the circuit for $N = 2$ is simply a Hadamard transform:

$$\mathcal{F}^\dagger_{\mathbb{Z}_2} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = H. \tag{144}$$

In the context of Shor's algorithm, we take N to be a power of 2, say $N = 2^n$, to represent a system of n qubits. Hence, we have n qubits, $N = 2^n$, and the inverse QFT we need is over $\mathbb{Z}_N$. So we can express j and k in (142) in terms of their binary representations:

$$j = j_{n-1}j_{n-2}\dots j_0, \quad k = k_{n-1}k_{n-2}\dots k_0, \qquad j_i, k_i \in \mathbb{F}_2. \tag{145}$$

Thus,

$$\mathcal{F}^\dagger_{\mathbb{Z}_N}|j\rangle = \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} \exp\left[-\frac{2\pi\iota}{2^n} \underbrace{\left(\sum_{\nu=0}^{n-1} k_\nu 2^\nu\right)}_{k} \underbrace{\left(\sum_{\mu=0}^{n-1} j_\mu 2^\mu\right)}_{j}\right]|k\rangle \tag{146}$$

$$= \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} \exp\left[-\frac{2\pi\iota}{2^n}\left(\sum_{\mu=0}^{n-1}\sum_{\nu=0}^{n-1} j_\mu k_\nu 2^{\mu+\nu}\right)\right]|k\rangle \tag{147}$$

$$= \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} \left(\prod_{\mu=0}^{n-1}\prod_{\nu=0}^{n-1} \exp\left[-2\pi\iota\left(j_\mu k_\nu 2^{\mu+\nu-n}\right)\right]\right)|k\rangle. \tag{148}$$

Note that if $j_\mu = k_\nu = 1$ and $\mu + \nu \geq n$, then $\exp[-2\pi\iota(j_\mu k_\nu 2^{\mu+\nu-n})] = 1$.

5.1.1 Two-qubit inverse QFT

$$n = 2 \quad \Rightarrow \quad \zeta_4 = -\iota \quad \Rightarrow \quad \mathcal{F}_{\mathbb{Z}_4}^\dagger = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -\iota & -1 & \iota \\ 1 & -1 & 1 & -1 \\ 1 & \iota & -1 & -\iota \end{bmatrix}. \tag{149}$$

The two qubits are originally in the basis state $|j_0 j_1\rangle$. There is some amplitude associated with them being in the basis state $|k_0 k_1\rangle$ after the inverse QFT.

$$\langle k|\mathcal{F}_{\mathbb{Z}_4}^\dagger|j\rangle = \frac{1}{2}\prod_{\mu=0}^{1}\prod_{\nu=0}^{1} \exp\left[-2\pi\iota\left(j_\mu k_\nu 2^{\mu+\nu-2}\right)\right] \tag{150}$$

$$= \frac{1}{2} e^{-2\pi\iota\frac{j_0 k_0}{4}} e^{-2\pi\iota\frac{j_0 k_1}{2}} e^{-2\pi\iota\frac{j_1 k_0}{2}} \underbrace{e^{-2\pi\iota j_1 k_1}}_{1} \tag{151}$$

$$= \frac{1}{2} e^{-2\pi\iota\frac{j_0 k_0}{4}} e^{-2\pi\iota\frac{j_0 k_1}{2}} e^{-2\pi\iota\frac{j_1 k_0}{2}} \tag{152}$$

$$= \frac{1}{2}(-\iota)^{j_0 k_0}(-1)^{j_0 k_1 + j_1 k_0} \tag{153}$$

$$= \frac{1}{2}(-\iota)^{j_0 k_0}(-1)^{\vec{j}\cdot\vec{k}'} \tag{154}$$

where $\vec{j} = (j_0, j_1)$ is the bit-string/vector ($\mathbb{Z}_2^2$) interpretation of j, and $\vec{k}' = \mathsf{SWAP}\left|\vec{k}\right\rangle$. Hence,

$$\langle k|\mathcal{F}_{\mathbb{Z}_4}^\dagger|j\rangle = (-\iota)^{j_0 k_0}\left\langle \vec{k}'\right| H \otimes H \left|\vec{j}\right\rangle \tag{155}$$

$$= (-\iota)^{j_0 k_0}\langle k_1 k_0|H^{\otimes 2}|j_0 j_1\rangle \tag{156}$$

$$= (-\iota)^{j_0 k_0}\langle k_1|H|j_0\rangle\langle k_0|H|j_1\rangle. \tag{157}$$

Now notice the key point: the phase $(-\iota)^{j_0 k_0}$ introduces a bilinear dependence between the input bit j_0 and the output bit k_0 of the Hadamard on $|j_1\rangle$, contributing a nontrivial phase only when both bits are 1. Hence, we apply a two-qubit unitary gate U to the two-qubit system when it's in the state $|j_0 k_0\rangle$ after the Hadamard on $|j_1\rangle$ to get $|j_0 k_0\rangle$ with the phase $(-\iota)^{j_0 k_0}$; and then follow it up with the Hadamard on $|j_0\rangle$. The question is what is the gate U such that $U|j_0 k_0\rangle = (-\iota)^{j_0 k_0}|j_0 k_0\rangle$? From this action, we can easily write down the matrix representation of U:

$$U = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & -\iota \end{bmatrix}. \tag{158}$$

This reveals that U is the controlled phase dagger gate: $\mathsf{C}S^\dagger$. Thus,

$$\langle k|\mathcal{F}^\dagger_{\mathbb{Z}_4}|j\rangle = \langle k_1k_0|H_1(\mathsf{C}S^\dagger)H_2|j_0j_1\rangle \tag{159}$$

$$= \langle k_0k_1|\mathsf{SWAP}(H_1)(\mathsf{C}S^\dagger)H_2|j_0j_1\rangle. \tag{160}$$

Since this is true for any basis states $|j\rangle$ and $|k\rangle$, we get gate equality

$$\mathcal{F}^\dagger_{\mathbb{Z}_4} = (\mathsf{SWAP})(H_1)(\mathsf{C}S^\dagger)(H_2). \tag{161}$$

Therefore, the two-qubit inverse QFT circuit is

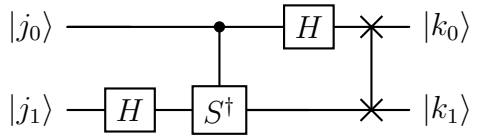

Figure 1: Two-qubit inverse QFT circuit.

5.1.2 n-qubit inverse QFT

The n qubits are originally in the basis state $|j_0j_1\ldots j_{n-1}\rangle$. There is some amplitude associated with them being in the basis state $|k_0k_1\ldots k_{n-1}\rangle$ after the inverse QFT.

$$\langle k|\mathcal{F}^\dagger_{\mathbb{Z}_N}|j\rangle = \frac{1}{\sqrt{2^n}}\prod_{\mu=0}^{n-1}\prod_{\nu=0}^{n-1}\exp\Big[-\frac{2\pi\iota}{2^n}(j_\mu k_\nu 2^{\mu+\nu})\Big]. \tag{162}$$

As we noted before, all the phases for which $\mu+\nu \geq n$ are 1 and vanish. Hence, we restrict the double product to only the "lower triangle:"

$$\langle k|\mathcal{F}^\dagger_{\mathbb{Z}_N}|j\rangle = \frac{1}{\sqrt{2^n}}\prod_{\mu=0}^{n-1}\prod_{\nu=0}^{n-\mu-1}\exp\Big[-\frac{2\pi\iota}{2^n}(j_\mu k_\nu 2^{\mu+\nu})\Big]. \tag{163}$$

Next, we factor out the part of the product where $\mu+\nu = n-1$.

$$\langle k|\mathcal{F}^\dagger_{\mathbb{Z}_N}|j\rangle = \frac{1}{\sqrt{2^n}}\left(\prod_{\substack{0\leq\mu,\nu<n-1\\ \mu+\nu<n-1}}\exp\Big[-\frac{2\pi\iota}{2^n}(j_\mu k_\nu 2^{\mu+\nu})\Big]\right)\left(\prod_{\tau=0}^{n-1}\exp\Big[-\frac{2\pi\iota}{2^n}(j_\tau k_{n-\tau-1}2^{n-1})\Big]\right) \tag{164}$$

$$= \frac{1}{\sqrt{2^n}}\left(\prod_{\substack{0\leq\mu,\nu<n-1\\ \mu+\nu<n-1}}\exp\Big[-\frac{2\pi\iota}{2^n}(j_\mu k_\nu 2^{\mu+\nu})\Big]\right)\left(\prod_{\tau=0}^{n-1}\exp[-\pi\iota(j_\tau k_{n-\tau-1})]\right) \tag{165}$$

$$= \frac{1}{\sqrt{2^n}}\left(\prod_{\substack{0\leq\mu,\nu<n-1\\ \mu+\nu<n-1}}\exp\Big[-\frac{2\pi\iota}{2^n}(j_\mu k_\nu 2^{\mu+\nu})\Big]\right)\exp\left[-\pi\iota\left(\sum_{\tau=0}^{n-1}j_\tau k_{n-\tau-1}\right)\right] \tag{166}$$

$$= \frac{1}{\sqrt{2^n}}\left(\prod_{\substack{0\leq\mu,\nu<n-1\\ \mu+\nu<n-1}}\exp\Big[-\frac{2\pi\iota}{2^n}(j_\mu k_\nu 2^{\mu+\nu})\Big]\right)(-1)^{\vec{j}\cdot\vec{k}'} \tag{167}$$

where $\vec{j} = (j_0, j_1, \ldots, j_{n-1})$ is the bit-string/vector ($\mathbb{Z}_2^n$) representation of j, and $\vec{k}' = (k_{n-1}, k_{n-2}, \ldots, k_0)$ is $\vec{k}$ in the opposite order. Hence,

$$\langle k|\mathcal{F}_{\mathbb{Z}_N}^\dagger|j\rangle = \left(\prod_{\substack{0\le\mu,\nu<n-1\\ \mu+\nu<n-1}} \exp\Big[-\frac{2\pi\iota}{2^n}(j_\mu k_\nu 2^{\mu+\nu})\Big]\right)\Big\langle\vec{k}'\Big|H^{\otimes n}\Big|\vec{j}\Big\rangle \tag{168}$$

$$= \left(\prod_{\substack{0\le\mu,\nu<n-1\\ \mu+\nu<n-1}} \exp\Big[-\frac{2\pi\iota}{2^n}(j_\mu k_\nu 2^{\mu+\nu})\Big]\right)\prod_{\tau=0}^{n-1}\langle k_{n-\tau-1}|H|j_\tau\rangle. \tag{169}$$

Now notice the key point: each phase factor in the product introduces a bilinear dependence between the input bit j_μ and the output bit k_ν of the Hadamard on $|j_{n-\nu-1}\rangle$, contributing a phase only when both bits are 1. Hence, we apply a two-qubit unitary $U_{\mu\nu}$ to the n-qubit system when it's in the state $\omega|j_0\ldots j_\mu\ldots j_{n-\nu-2}k_\nu\ldots k_0\rangle$, where ω is some phase, so that it pops out an additional phase of $\zeta_{2^n}^{j_\mu k_\nu 2^{\mu+\nu}}$. Since this phase is nontrivial only when $j_\mu = k_\nu = 1$, it can be realized by a controlled phase gate with control on qubit μ and target on qubit ν. The right gate that does this is the controlled $R_{n-(\mu+\nu)}$ gate, where

$$R_\alpha = \begin{bmatrix} 1 & 0 \\ 0 & \zeta_{2^\alpha} \end{bmatrix}. \tag{170}$$

When both control and target qubits are in the state $|j_\mu\rangle = |k_\nu\rangle = |1\rangle$, this gate contributes the phase

$$\zeta_{2^{n-(\mu+\nu)}} = \exp\Big(-\frac{2\pi\iota}{2^{n-(\mu+\nu)}}\Big) = \exp\Big(-\frac{2\pi\iota}{2^n}2^{\mu+\nu}\Big) = \zeta_{2^n}^{2^{\mu+\nu}} \tag{171}$$

which matches the required phase factor. Therefore, after applying the Hadamard to qubit $n-\nu-1$ in state $|j_{n-\nu-1}\rangle$ to produce state $|k_\nu\rangle$, we apply controlled $R_{n-(\mu+\nu)}$ gates with control on qubit μ (in state $|j_\mu\rangle$) and target on qubit $n-\nu-1$ (in state $|k_\nu\rangle$) for all $0 \le \mu \le n-\nu-2$.

As an example, the six-qubit inverse QFT circuit is drawn below. It clearly demonstrates the pattern that applies for general n.

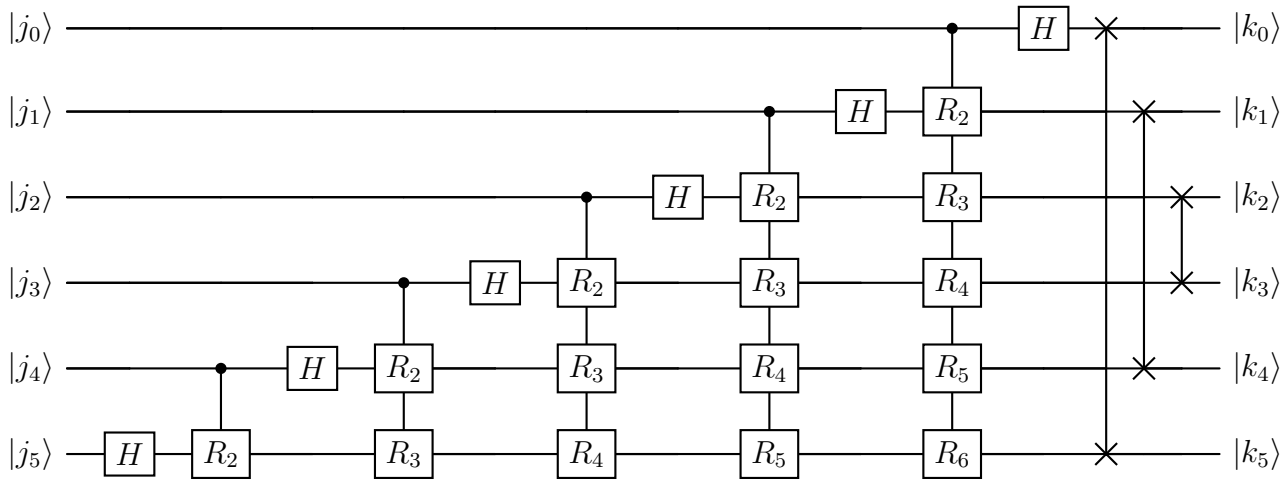

Figure 2: Six-qubit inverse QFT circuit.

In the n-qubit inverse QFT circuit, there are

- n single-qubit (Hadamard) gates,
- $\binom{n}{2} \approx \frac{n^2}{2}$ two-qubit ($\mathsf{C}R_\alpha$) gates.

The larger the value of α, the closer is the $\mathsf{C}R_\alpha$ gate to the identity (rigorous argument using process fidelity metric). Hence, the inverse QFT becomes insensitive to dropping small rotations; i.e., $\mathsf{C}R_\alpha$ gates with large αs.

5.2 Quantum Algorithm for Periodicity

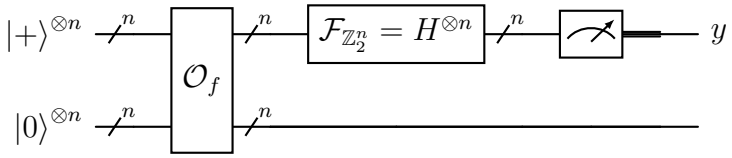

Figure 3: Quantum circuit for Simon's algorithm.

Suppose we have a (black box) periodic function f over $\mathbb{Z}$ with period r; i.e., such that

$$f(x) = f(x+r) = f(x+2r) = \dots. \tag{172}$$

Then, a classical algorithm takes $\Theta(r)$ time to find r, but it is possible to find r using a quantum algorithm in polylogarithmic time. What is the quantum algorithm for this? Before we answer that question, we must specify some more constraints on f. Suppose

- *Domain restriction:* The domain of f is restricted to integers that can be expressed in binary using at most $2n$ bits, where n is some nonnegative integer; i.e.,

$$\mathsf{dom}\, f = \Big\{0, \ldots, 2^{2n} - 1\Big\}. \tag{173}$$

- *Range restriction:* The range of f is restricted to values that can be expressed in binary using at most n bits. Thus,

$$f : \{0,1\}^{2n} \to \{0,1\}^{n}. \tag{174}$$

- *Period bound:* The period r is upper-bounded by some $r_{\max}$—i.e., $r_{\max} \geq r$—such that

$$2^n \geq 2r_{\max}. \tag{175}$$

Let $\mathcal{O}_f$ be the oracle

$$|x\rangle|y\rangle \longrightarrow \boxed{\mathcal{O}_f} \longrightarrow |x\rangle|y \oplus f(x)\rangle \tag{176}$$

which is the same as the one used in Simon's algorithm. Then, consider the following circuit:

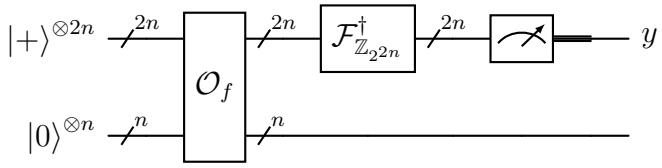

Figure 4: Quantum circuit for order-finding.

5.2.1 Analysis

Let $N = 2^n$. Start with

$$|+\rangle^{\otimes 2n}|0\rangle^{\otimes n} = \frac{1}{N}\sum_{x=0}^{N^2-1}|x\rangle|0^n\rangle. \tag{177}$$

Compute f:

$$\mathcal{O}_f|+\rangle^{\otimes 2n}|0\rangle^{\otimes n} = \frac{1}{N}\sum_{x=0}^{N^2-1}|x\rangle|f(x)\rangle. \tag{178}$$

Then, take the inverse QFT over $\mathbb{Z}_{N^2}$:

$$\Big(\mathcal{F}^{\dagger}_{\mathbb{Z}_{N^2}} \otimes I\Big)\mathcal{O}_f|+\rangle^{\otimes 2n}|0\rangle^{\otimes n} = \frac{1}{N^2}\sum_{x=0}^{N^2-1}\sum_{y=0}^{N^2-1}\exp\Big(-\frac{2\pi\iota}{N^2}xy\Big)|y\rangle|f(x)\rangle. \tag{179}$$

Now measure the first $2n$ qubits, i.e. y. Let y_0 be the measurement outcome. Then, the amplitude of $|y_0\rangle$ in (179) is

$$|C_{y_0}\rangle = \frac{1}{N^2} \sum_{x=0}^{N^2-1} \zeta_{N^2}^{xy} |f(x)\rangle. \tag{180}$$

Hence,

$$\begin{aligned}
\mathbb{P}(y_0) &= \| |C_{y_0}\rangle \|^2 && (181)\\
&= \langle C_{y_0} | C_{y_0} \rangle && (182)\\
&= \frac{1}{N^4} \sum_{x'=0}^{N^2-1} \sum_{x=0}^{N^2-1} \zeta_{N^2}^{(x-x')y} \langle f(x') | f(x) \rangle. && (183)
\end{aligned}$$

The inner product $\langle f(x')|f(x)\rangle$ is 1 iff $f(x') = f(x)$; i.e., iff x and x' differ by an integer multiple of the period r. Otherwise, the inner product is 0. Keeping this in mind, we can rewrite the double sum in (183) in a different way. First, we split up the possible xs into equivalence classes under the relation $x \sim x' \Leftrightarrow f(x) = f(x')$. Hence, we get r such classes, with the elements in each class forming an arithmetic progression with common difference r. Suppose the smallest element in some class is x_0. Then, the class is

$$[x_0] = \{x_0, x_0 + r, x_0 + 2r, \ldots, x_0 + (J_{x_0} - 1)r\} \tag{184}$$

where J_{x_0} determines the largest element in $[x_0]$. Since there are N^2 elements and r classes,

$$J_{x_0} \in \left\{ \left\lfloor \frac{N^2}{r} \right\rfloor, \left\lceil \frac{N^2}{r} \right\rceil \right\}. \tag{185}$$

Note that $|[x_0]| = J_{x_0}$. More precisely, suppose, by the division algorithm, that $N^2 = ar + b$ for quotient a and remainder b; then,

$$J_{x_0} = \begin{cases} a+1, & 0 \le x_0 < b, \\ a, & b \le x_0 < r. \end{cases} \tag{186}$$

Using this notation, we can rewrite the double sum in (183) as

$$\begin{aligned}
\mathbb{P}(y_0) &= \frac{1}{N^4} \sum_{x_0=0}^{r-1} \sum_{z'=0}^{J_{x_0}-1} \sum_{z=0}^{J_{x_0}-1} \zeta_{N^2}^{(z-z')ry_0} && (187)\\
&= \frac{1}{N^4} \sum_{x_0=0}^{r-1} \sum_{z'=0}^{J_{x_0}-1} \sum_{z=0}^{J_{x_0}-1} \zeta_{N^2}^{zry_0} \left(\zeta_{N^2}^{z'ry_0}\right)^* && (188)\\
&= \frac{1}{N^4} \sum_{x_0=0}^{r-1} \left[\sum_{z=0}^{J_{x_0}-1} \zeta_{N^2}^{zry_0} \right] \left[\sum_{z'=0}^{J_{x_0}-1} \left(\zeta_{N^2}^{z'ry_0}\right)^* \right] && (189)\\
&= \frac{1}{N^4} \sum_{x_0=0}^{r-1} \left(\sum_{z=0}^{J_{x_0}-1} \zeta_{N^2}^{zry_0} \right) \left(\sum_{z'=0}^{J_{x_0}-1} \zeta_{N^2}^{z'ry_0} \right)^* && (190)
\end{aligned}$$

$$= \frac{1}{N^4} \sum_{x_0=0}^{r-1} \left| \sum_{z=0}^{J_{x_0}-1} \zeta_{N^2}^{zry_0} \right|^2. \tag{191}$$

The z-sum is nothing but the sum of the first J_{x_0} terms of the geometric progression with first term 1 and common ratio $\zeta_{N^2}^{ry_0}$. Hence,

$$S_{x_0y_0} \doteq \sum_{z=0}^{J_{x_0}-1} (\zeta_{N^2}^{ry_0})^z = \begin{cases} J_{x_0} & \text{if } \zeta_{N^2}^{ry_0} = 1 \Leftrightarrow \frac{ry_0}{N^2} \in \mathbb{Z}, \\ \frac{1-\zeta_{N^2}^{J_{x_0} ry_0}}{1-\zeta_{N^2}^{ry_0}} & \text{if } \zeta_{N^2}^{ry_0} \neq 1 \Leftrightarrow \frac{ry_0}{N^2} \notin \mathbb{Z}. \end{cases} \quad \text{and} \quad \mathbb{P}(y_0) = \frac{1}{N^4} \sum_{x_0=0}^{r-1} |S_{x_0y_0}|^2. \tag{192}$$

The exact case $\frac{ry_0}{N^2} \in \mathbb{Z}$ gives perfectly aligned phases and probability $\mathbb{P}(y_0) \approx \frac{1}{r}$ per peak. However, in general, r need not divide N^2, so exact peaks are not the main success event. The important event is that $\frac{ry_0}{N^2}$ is close to an integer. Assume $\frac{ry_0}{N^2} \notin \mathbb{Z}$. Let $\theta_{y_0} = \frac{ry_0}{N^2}$. Let the closest integer to θ_{y_0} be d_{y_0}. Let $\delta_{y_0} \doteq |d_{y_0} - \theta_{y_0}|$. Then,

$$\mathbb{P}(y_0) = \frac{1}{N^4} \sum_{x_0=0}^{r-1} \frac{\left|1 - e^{-2\pi\iota J_{x_0}\theta_{y_0}}\right|^2}{\left|1 - e^{-2\pi\iota\theta_{y_0}}\right|^2} \tag{193}$$

$$= \frac{1}{N^4} \sum_{x_0=0}^{r-1} \frac{\sin^2(\pi J_{x_0}\theta_{y_0})}{\sin^2(\pi\theta_{y_0})} \tag{194}$$

using the fact that $|1 - e^{-\iota\alpha}| = 2\left|\sin\frac{\alpha}{2}\right|$. Since θ_{y_0} is within distance δ_{y_0} of the integer d_{y_0}, we have

$$\sin^2(\pi\theta_{y_0}) = \sin^2(\pi\delta_{y_0}). \tag{195}$$

To concretize the "near-integer" ("main peak") region, suppose

$$\delta_{y_0} \leq \frac{1}{2J_{x_0}} \approx \frac{r}{2N^2}. \tag{196}$$

Then, $\pi J_{x_0}\delta_{y_0} \leq \frac{\pi}{2}$. Hence,

$$\sin(\pi J_{x_0}\theta_{y_0}) = \sin(\pi J_{x_0}\delta_{y_0}) \geq \frac{2}{\pi}(\pi J_{x_0}\delta_{y_0}) = 2J_{x_0}\delta_{y_0}. \tag{197}$$

Moreover, since $0 < \delta_{y_0} \lesssim \frac{r}{2N^2} \leq \frac{1}{2}$,

$$\sin(\pi\delta_{y_0}) \leq \pi\delta_{y_0}. \tag{198}$$

Thus, from (197) and (198),

$$\frac{\sin^2(\pi J_{x_0}\theta_{y_0})}{\sin^2(\pi\theta_{y_0})} = \frac{\sin^2(\pi J_{x_0}\delta_{y_0})}{\sin^2(\pi\delta_{y_0})} \geq \frac{4J_{x_0}^2}{\pi^2} \approx \frac{4N^4}{\pi^2 r^2}. \tag{199}$$

Therefore, for near-integer outcomes y_0 (i.e., those satisfying $\delta_{y_0} \lesssim \frac{r}{2N^2}$), we have

$$\mathbb{P}(y_0) \gtrsim \frac{4}{\pi^2 r} \approx \frac{0.4053}{r}. \tag{200}$$

5.2.1.1 Summary Finally, to paint the complete picture, if we plot $\mathbb{P}(y)$ vs. y for $0 \leq y \leq N^2 - 1$, then we observe spikes at $y = 0, \frac{N^2}{r}, \frac{2N^2}{r}, \ldots, \frac{(r-1)N^2}{r}$ since these are the values at which $\frac{ry}{N^2} \in \mathbb{Z}$ and $\mathbb{P}(y) \sim \frac{1}{r}$. Hence, we observe spikes at "exact-integer" outcomes. The probability decays as inverse-square in the distance from the nearest spike. This can be shown by applying the following bounds:

$$\sin(\pi\delta_y) \geq 2\delta_y \quad \text{and} \quad \sin(\pi J_{x_0}\delta_y) \leq 1. \tag{201}$$

Define $\Delta_y = \left|y - \frac{d_y N^2}{r}\right|$ as the distance from the nearest spike. Then, $\delta_y = \frac{r}{N^2}\Delta_y$. Hence, from the bounds,

$$\frac{\sin^2(\pi J_{x_0}\delta_y)}{\sin^2(\pi\delta_y)} \leq \frac{1}{4\delta_y^2} \tag{202}$$

$$\Rightarrow \mathbb{P}(y) \leq \frac{r}{4N^4\delta_y^2} = \frac{1}{4r\Delta_y^2}. \tag{203}$$

If we restrict our focus to "near-integer" values defined by the condition $\Delta_y \lesssim \frac{1}{2}$, then we get that the probability of obtaining an outcome within $\frac{1}{2}$-step of a spike is at least

$$\mathbb{P}(y) \geq \frac{4}{\pi^2 r}. \tag{204}$$

Since there are exactly r spikes, the total probability of landing near some spike is at least

$$r \cdot \frac{4}{\pi^2 r} = \frac{4}{\pi^2} \approx 0.4053. \tag{205}$$

Therefore, in one run of the order-finding algorithm,

$$\mathbb{P}\Big(y \text{ within } \frac{1}{2}\text{-step of a spike}\Big) = \Theta(1). \tag{206}$$

A valid question at this point would be: "but how does all this help in actually finding the order r, which we still have not done?" Indeed, the quantum order-finding algorithm does not directly output the order r. Rather, its purpose is to transform the hidden periodic structure of f into measurable frequency information. Concretely, the goal is to obtain a measurement outcome y such that $\theta_y = \frac{ry}{N^2}$ lies close to an integer d_y. Equivalently, we seek a value of y lying near one of the probability spikes analyzed above. Although a single execution does not guarantee such an outcome, the preceding analysis shows that it occurs with constant probability. Consequently, polynomially many repetitions suffice to make the probability of never obtaining a useful near-peak outcome exponentially small. Once such a value is obtained, the ratio $\frac{y}{N^2}$ provides an accurate rational approximation to $\frac{d_y}{r}$, from which the order r can then be efficiently recovered using classical post-processing viz. the continued fractions algorithm.

5.3 Continued Fractions and Recovery of the Period

The quantum order-finding subroutine gives us an integer y such that $0 \leq y \leq N^2 - 1$ and

$$\frac{y}{N^2} \approx \frac{d}{r} \tag{207}$$

for some integer d. The remaining question is how to recover the unknown denominator r from this approximation. This is where *continued fractions* come in. A continued fraction is an expression of the form

$$[a_0; a_1, a_2, \dots] = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cdots}}, \tag{208}$$

where $a_0 \in \mathbb{Z}$ and $a_i \in \mathbb{Z}^+$ for $i \geq 1$. Every rational number has a finite continued fraction expansion. The finite truncations

$$[a_0; a_1, a_2, \dots, a_k] \tag{209}$$

are called *convergents*. These convergents are especially good rational approximations to the original number. The algorithm itself is as follows. Given a real number x, repeatedly take integer parts and reciprocals:

$$a_0 = \lfloor x \rfloor, \qquad x_1 = \frac{1}{x - a_0}, \tag{210}$$

then

$$a_1 = \lfloor x_1 \rfloor, \qquad x_2 = \frac{1}{x_1 - a_1}, \tag{211}$$

and so on. This produces the continued fraction expansion

$$x = [a_0; a_1, a_2, \dots]. \tag{212}$$

At each stage, compute the convergent

$$\frac{p_k}{q_k} = [a_0; a_1, a_2, \dots, a_k]. \tag{213}$$

The denominators q_k increase, so in Shor's algorithm we simply compute convergents until the denominator exceeds the allowed bound (175) for r. We finally use the following standard theorem from continued fractions theory.

Theorem (Continued fractions recovery theorem)**.** Let $x \in \mathbb{R}$ and let $\frac{a}{b}$ be a rational number in lowest terms. If

$$\left| x - \frac{a}{b} \right| < \frac{1}{2b^2}, \tag{214}$$

then $\frac{a}{b}$ appears as one of the convergents in the continued fraction expansion of x.

5.3.1 Application to order finding

From the Fourier sampling analysis, with constant probability the measured value y satisfies

$$\left|\frac{y}{N^2} - \frac{d}{r}\right| \lesssim \frac{1}{2N^2}. \tag{215}$$

From (175), Shor chooses N large enough that $N^2 \geq 4r^2$. Hence, $\frac{1}{2N^2} < \frac{1}{2r^2}$. Therefore,

$$\left|\frac{y}{N^2} - \frac{d}{r}\right| < \frac{1}{2r^2}. \tag{216}$$

This is exactly the regime where the continued fractions recovery theorem applies. Therefore, $\frac{d}{r}$ appears among the convergents of the continued fraction expansion of $\frac{y}{N^2}$.

5.3.2 One caveat

A subtlety remains: the fraction $\frac{d}{r}$ need not be in lowest terms. If $g = \gcd(d, r)$, then $\frac{d}{r} = \frac{d'}{r'}$ where $d' = \frac{d}{g}$, $r' = \frac{r}{g}$, and $\gcd(d', r') = 1$. The continued fractions algorithm therefore recovers the reduced denominator r', which is a divisor of the true order r. However, this does not pose a problem. Repeating the quantum order-finding procedure yields independent values of d, and with non-negligible probability we obtain a sample satisfying $\gcd(d, r) = 1$. Indeed, among the integers $0 \leq d < r$, exactly $\varphi(r)$ values are coprime to r, where φ denotes Euler's totient function. Hence,

$$\mathbb{P}(\gcd(d, r) = 1) = \frac{\varphi(r)}{r}. \tag{217}$$

Using the standard bound

$$\frac{\varphi(r)}{r} = \Omega\left(\frac{1}{\log\log r}\right), \tag{218}$$

the probability of obtaining a coprime numerator is inverse-polylogarithmic in r, and thus non-negligible. In that case, the reduced denominator equals the true order r, and the algorithm succeeds. Since each run of the quantum order-finding subroutine produces a useful near-spike outcome with constant probability, polynomially many repetitions suffice to recover the order r with high probability.

5.4 Reduction of Factoring to Order Finding

Suppose we have $N = pq$ for distinct odd prime p and q. The trick is to find x and y such that

$$x^2 \equiv y^2 \bmod N \quad \text{but} \quad x \not\equiv \pm y \bmod N. \tag{219}$$

Then,

$$(x + y)(x - y) \equiv 0 \bmod N. \tag{220}$$

Hence, N divides the product $(x + y)(x - y)$ without dividing either factor individually. Therefore $\gcd(x + y, N)$ or $\gcd(x - y, N)$ computed using the Euclidean algorithm yields a nontrivial factor. This congruence-of-squares framework underlies several factoring algorithms, including the quadratic sieve and Shor's algorithm.

5.4.1 How do we find such x, y?

Look at the function

$$f(x) = g^x \bmod N, \tag{221}$$

where $g \in \mathbb{Z}_N^\times$. This is periodic because if $g^r \equiv 1 \bmod N$, then $g^{x+r} \equiv g^x \bmod N$. Hence, $f(x)$ is periodic with period r. Thus, if r is even, then

$$\underbrace{\left(g^{\frac{r}{2}}\right)^2}_{x^2} \equiv \underbrace{1^2}_{y^2} \bmod N. \tag{222}$$

5.4.2 How can this fail?

This may go wrong if:

- The order r of our chosen g is odd.
- $g^{\frac{r}{2}} \equiv -1 \bmod N$.

However, we rely on the following theorem.

Theorem. For a randomly chosen $g \in \mathbb{Z}_N^\times$,

$$\mathbb{P}(\text{success}) \geq \frac{1}{2}. \tag{223}$$

5.5 Runtime Complexity

The quantum part of Shor's algorithm consists primarily of two tasks:

1. Modular exponentiation: $|x\rangle|0\rangle \mapsto |x\rangle|g^x \bmod N\rangle$, where $g, x < N$. This is needed to implement the oracle $\mathcal{O}_{g^x \bmod N}$.
2. Inverse QFT on n qubits: This is needed in the quantum order-finding subroutine.

5.5.1 Efficient modular exponentiation

The obvious way to do modular exponentiation is repeated multiplication by g. But this is really inefficient because it must be done x times, and x can be large if N is large. The efficient algorithm is *repeated squaring*. We classically precompute

$$g \bmod N, g^2 \bmod N, g^4 \bmod N, g^8 \bmod N, \ldots. \tag{224}$$

Then, we express x in binary, and do a controlled g^{2^i} gate with control on the i^{th} bit for all $0 \leq i < n$.

This requires $O(n)$ multiplications (mod$N$) since x has n bits, where each multiplication is between two n-bit numbers. Hence, using longhand (kindergarten) multiplication, each multiplication takes $O(n^2)$ time. Therefore, modular exponentiation takes $O(n^3)$ time. This can indeed be done reversibly using Toffoli gates.

5.5.2 iQFT complexity

The inverse QFT on n qubits requires $O(n^2)$ elementary quantum gates, or $O(n \log n)$ gates under standard approximation schemes in which sufficiently small controlled rotations are omitted.

5.5.3 Overall runtime

The $O(n^3)$ time of modular exponentiation dominates the $O(n^2)$ time of the inverse QFT, making the overall runtime of Shor's factoring algorithm for factoring an n-bit number N polynomial in $\lg N$: $O(\lg^3 N)$.

6 Quantum Phase Estimation

Suppose we have a unitary U and an eigenvector $|\psi\rangle$, and suppose we can compute U^k easily. Then, since U is unitary, we know that

$$U|\psi\rangle = e^{\iota\theta}|\psi\rangle \tag{225}$$

for some θ.

Problem: Estimate θ.

6.1 Quantum Algorithm

1. Start with an n-qubit phase register initialized to $|0\rangle^{\otimes n}$, and a second register initialized to the eigenstate $|\psi\rangle$.
2. Apply $H^{\otimes n}$ on the first register to get a superposition of all n-bit integers:

$$(H^{\otimes n} \otimes I)|0^n\rangle|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} |k\rangle|\psi\rangle. \tag{226}$$

3. Apply n controlled gates as follows: We apply a controlled $U^{2^{i-1}}$ gate with control on the i^{th} qubit in the first register (1-indexed) and target on the second register. Note that a U^j gate on the second register just kicks back a phase of $e^{\iota j\theta}$. Also note that this sequence of controlled gates is identical to applying a U^k gate when the first n qubits are in the state $|k\rangle$, $k \in \mathbb{Z}$. Hence,

$$(\mathsf{C}U^{2^{n-1}} \dots \mathsf{C}U)(H^{\otimes n} \otimes I)|0^n\rangle|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} |k\rangle(U^k|\psi\rangle) \tag{227}$$

$$= \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{\iota k\theta}|k\rangle|\psi\rangle. \tag{228}$$

4. Apply an inverse QFT $\mathcal{F}^{\dagger}_{\mathbb{Z}_{2^n}}$ to the first register.

$$\left(\mathcal{F}^{\dagger}_{\mathbb{Z}_{2^n}} \otimes I\right)(\mathsf{C}U^{2^{n-1}} \dots \mathsf{C}U)(H^{\otimes n} \otimes I)|0^n\rangle|\psi\rangle = \frac{1}{2^n}\sum_{k=0}^{2^n-1} e^{\iota k\theta}\sum_{\ell=0}^{2^n-1} e^{-\frac{2\pi\iota}{2^n}k\ell}|\ell\rangle|\psi\rangle \tag{229}$$

$$= \frac{1}{2^n}\sum_{k=0}^{2^n-1}\sum_{\ell=0}^{2^n-1} e^{\iota k\left(\theta-\frac{2\pi\ell}{2^n}\right)}|\ell\rangle|\psi\rangle. \tag{230}$$

5. Measure ℓ.

6.2 Analysis

For a fixed measurement outcome ℓ_0, the amplitude of $|\ell\rangle_0$ is

$$\frac{1}{2^n}\sum_{k=0}^{2^n-1}\exp\left[\iota k\left(\theta-\frac{2\pi\ell_0}{2^n}\right)\right]. \tag{231}$$

This is precisely the same type of geometric sum that appeared in the analysis of Shor's algorithm. In particular, it plays the same role as the object

$$S_{x_0y_0} = \sum_{z=0}^{J_{x_0}-1}\exp\left(-\frac{2\pi\iota}{N^2}ry_0z\right) \tag{232}$$

arising in the order-finding analysis. In both cases, the amplitude is obtained by summing complex phases with a fixed phase increment between consecutive terms, which is precisely the common ratio of the geometric progression. In other words, the consecutive terms differ by a fixed angular increment on the complex unit circle. Here, the increment is $\theta - \frac{2\pi\ell_0}{2^n}$, whereas in the case of Shor's algorithm, it is $\frac{2\pi ry_0}{2^{2n}}$.

As k increases, the successive terms therefore trace out points around the unit circle separated by this fixed angle. If the increment is far from an integer multiple of 2π, the phases wind around the circle and tend to cancel each other out, producing a small net amplitude. However, if the increment is close to an integer multiple of 2π, successive terms point in nearly the same direction, so the contributions add coherently rather than destructively. This constructive interference produces a large amplitude and therefore a large measurement probability. Consequently, the most likely outcomes are those satisfying

$$\theta - \frac{2\pi\ell}{2^n} \approx 0; \tag{233}$$

i.e., $\theta \approx \frac{2\pi\ell}{2^n}$. Thus, the inverse QFT converts the phase information encoded in the amplitudes into localization of probability near the computational basis state corresponding to the closest discretized approximation of the eigenphase θ.

6.3 Aside: Fourier Transform as Phase Alignment

The phenomenon underlying both quantum phase estimation and Shor's algorithm is not specific to quantum computing; it is the fundamental mechanism behind Fourier analysis

itself. A Fourier transform compares a signal against complex phases that move around the unit circle at different angular speeds. For a fixed frequency, one repeatedly takes equal angular steps around the circle and adds the resulting complex numbers together. If the signal contains oscillatory structure with the same angular progression, the phases remain nearly aligned and add coherently, producing a large Fourier coefficient. If the frequency does not match, the phases point in many different directions around the circle and largely cancel out.

In this sense, a Fourier transform converts hidden oscillatory structure into concentration in frequency space. For example, if a discrete signal contains a periodic component with frequency ω_0, then the Fourier transform becomes large near ω_0 because the corresponding complex phases stay approximately aligned across the sum. Quantum phase estimation and Shor's algorithm exploit exactly the same idea: the QFT detects the phase progression already present in the amplitudes and concentrates measurement probability near the matching frequency.

6.4 Factoring using QPE

In the factoring problem, we take the unitary U given by

$$|x\rangle \mapsto |gx \bmod N\rangle \tag{234}$$

for some $g \in \mathbb{Z}_N^\times$, where N is the number to be factored. Suppose g has order r; i.e., $g^r \equiv 1 \pmod N$.

Claim. $|\psi_\ell\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \exp\left(-\frac{2\pi\iota k\ell}{r}\right)\left|g^k\right\rangle$ is an eigenvector of U.

Proof. Compute:

$$\begin{aligned}
U|\psi_\ell\rangle &= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \exp\left(-\frac{2\pi\iota k\ell}{r}\right) U\left|g^k\right\rangle && (235)\\
&= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \exp\left(-\frac{2\pi\iota k\ell}{r}\right) \left|g^{k+1}\right\rangle && (236)\\
&= \frac{1}{\sqrt{r}} \sum_{k'=1}^{r} \exp\left(-\frac{2\pi\iota (k'-1)\ell}{r}\right) \left|g^{k'}\right\rangle && (237)\\
&= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \exp\left(-\frac{2\pi\iota (k-1)\ell}{r}\right) \left|g^{k}\right\rangle \qquad (\because g^0 \equiv g^r \equiv 1 \pmod N) && (238)\\
&= \exp\left(\frac{2\pi\iota\ell}{r}\right) \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \exp\left(-\frac{2\pi\iota k\ell}{r}\right) \left|g^{k}\right\rangle && (239)\\
&= \exp\left(\frac{2\pi\iota\ell}{r}\right) |\psi_\ell\rangle. && (240)
\end{aligned}$$

□

Therefore, the corresponding eigenvalue is

$$\exp\left(\frac{2\pi\iota\ell}{r}\right), \tag{241}$$

so the corresponding eigenphase is

$$\theta \equiv \frac{2\pi\ell}{r} \pmod{2\pi}. \tag{242}$$

Hence, QPE estimates

$$\frac{\theta}{2\pi} \equiv \frac{\ell}{r} \pmod{1}. \tag{243}$$

Equivalently, the measured phase is a rational number with denominator r. Applying the continued fractions algorithm to this phase recovers r.

Note: In the order-finding procedure, one does not explicitly prepare a specific eigenstate $|\psi_\ell\rangle$. Indeed, preparing a chosen state $|\psi_\ell\rangle$ would require knowledge of the corresponding value of ℓ, which is itself determined by the unknown order r. Instead the state $|1\rangle = |g^0\rangle$ is an equal superposition of these eigenstates, so QPE samples one of the eigenphases $\sim\frac{\ell}{r}$ at random.

7 Grover's Search Algorithm

Problem 1: Suppose we have an unsorted list of N things in quantum memory, and one of the items is "marked." Find the marked item.

For this problem as stated, Grover's algorithm takes $O\left(\sqrt{N}\right)$ time, which is much slower than what we could do with a real classical database. Hence, this motivating problem sells Grover's algorithm short. However, Grover's algorithm works for finding "implicit lists."

Problem 2: Suppose we want to find w, x, y, z with

$$x^4 + y^4 + z^4 = w^4, \tag{244}$$

and we know that there exists a solution with $w, x, y, z < 10^6$.

Classical time by brute-force: We pick random values for x, y, z, take the fourth root of $x^4 + y^4 + z^4$ and check whether the result is an integer. Hence, we search through a list of 10^{18} triples, which takes $O(10^{18})$ time.

Grover's algorithm takes $O(10^9)$ time.

7.1 Quantum Algorithm

In the setup of problem 2, assume there exists only one solution in N triples; i.e., there is only a single marked item out of N of them. Let ϕ be a bijective "indexing" map from the set of N triples to the set of whole numbers from 0 through $N-1$. If (x_0, y_0, z_0) is the solution, then let $\beta = \phi(x_0, y_0, z_0)$. Henceforth, we work in terms of the indices. Assume we have a (black-box unitary) phase oracle $\mathcal{O}$ such that

$$|x\rangle \longrightarrow \boxed{\mathcal{O}} \longrightarrow (-1)^{\delta_{x\beta}}|x\rangle. \tag{245}$$

Also assume $N = 2^n$ for simplicity (wlog).

7.1.1 Grover iteration

We start with n qubits initialized to $|+\rangle^{\otimes n}$. One Grover iteration consists of four steps:

1. Apply $\mathcal{O}$.
2. Apply $H^{\otimes n}$.
3. Apply $|x\rangle \mapsto -(-1)^{\delta_{x0}}|x\rangle$.
4. Apply $H^{\otimes n}$.

The number of Grover iterations needed to get the solution with high probability is $\left\lfloor \frac{\pi}{4}\sqrt{N} \right\rceil$.

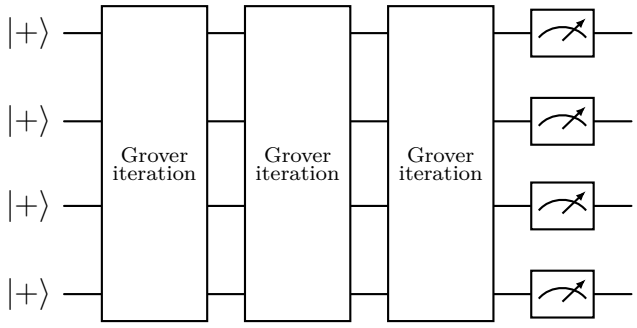

Figure 5: Quantum circuit for Grover's algorithm with $N = 16$.

7.1.2 Analysis

Let

$$|\alpha\rangle = \frac{1}{\sqrt{N-1}} \sum_{x \neq \beta} |x\rangle. \tag{246}$$

In the first Grover iteration, we start with

$$|+\rangle^{\otimes n} = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle \tag{247}$$

$$= \sqrt{\frac{N-1}{N}}|\alpha\rangle + \frac{1}{\sqrt{N}}|\beta\rangle. \tag{248}$$

Hence,

$$\mathcal{O}|+\rangle^{\otimes n} = \sqrt{\frac{N-1}{N}}|\alpha\rangle - \frac{1}{\sqrt{N}}|\beta\rangle. \tag{249}$$

Next, we construct the operator for the next three steps of a Grover iteration. A matrix that does the step 3 of the Grover iteration is

$$2|0\rangle\langle 0| - I_N = \begin{bmatrix} 1 & & & & \\ & -1 & & & \\ & & -1 & & \\ & & & \ddots & \\ & & & & -1 \end{bmatrix}. \tag{250}$$

Hence,

$$H^{\otimes n}(2|0\rangle\langle 0| - I_N)H^{\otimes n} = 2|+^n\rangle\langle +^n| - I_N \tag{251}$$

$$= \frac{2}{N}\Big(\sqrt{N-1}|\alpha\rangle + |\beta\rangle\Big)\Big(\sqrt{N-1}\langle\alpha| + \langle\beta|\Big) - I_N \tag{252}$$

$$= \frac{2}{N}\Big[(N-1)|\alpha\rangle\langle\alpha| + \sqrt{N-1}|\alpha\rangle\langle\beta| + \sqrt{N-1}|\beta\rangle\langle\alpha| + |\beta\rangle\langle\beta|\Big] - I_N. \tag{253}$$

Note that $\langle\alpha|\beta\rangle = 0$ and $\langle\alpha|\alpha\rangle = \langle\beta|\beta\rangle = 1$. Therefore,

$$H^{\otimes n}(2|0\rangle\langle 0| - I_N)H^{\otimes n}\mathcal{O}|+^n\rangle = \frac{2}{N}\sqrt{\frac{N-1}{N}}\Big[(N-1)|\alpha\rangle + \sqrt{N-1}|\beta\rangle\Big] \tag{254}$$

$$- \frac{2}{N\sqrt{N}}\Big[\sqrt{N-1}|\alpha\rangle + |\beta\rangle\Big] \tag{255}$$

$$- \sqrt{\frac{N-1}{N}}|\alpha\rangle + \frac{1}{\sqrt{N}}|\beta\rangle \tag{256}$$

$$= c_\alpha|\alpha\rangle + c_\beta|\beta\rangle, \tag{257}$$

where

$$c_\alpha = \frac{2(N-1)\sqrt{N-1}}{N\sqrt{N}} - \frac{2\sqrt{N-1}}{N\sqrt{N}} - \sqrt{\frac{N-1}{N}} \tag{258}$$

$$= \frac{2(N-1)\sqrt{N-1} - 2\sqrt{N-1} - N\sqrt{N-1}}{N\sqrt{N}} \tag{259}$$

$$= \frac{(N-4)\sqrt{N-1}}{N\sqrt{N}}, \tag{260}$$

and

$$c_\beta = \frac{2(N-1)}{N\sqrt{N}} - \frac{2}{N\sqrt{N}} + \frac{1}{\sqrt{N}} \tag{261}$$

$$= \frac{3N-4}{N\sqrt{N}}. \tag{262}$$

Suppose $\mathcal{G}$ is the unitary for one Grover iteration; i.e., $\mathcal{G} = H^{\otimes n}(2|0\rangle\langle 0| - I_N)H^{\otimes n}\mathcal{O}$. Thus,

$$\mathcal{G}|+^n\rangle = \frac{(N-4)\sqrt{N-1}}{N\sqrt{N}}|\alpha\rangle + \frac{3N-4}{N\sqrt{N}}|\beta\rangle. \tag{263}$$

In general, if we start with the normalized state $a|\alpha\rangle + b|\beta\rangle$, then

$$\mathcal{G}(a|\alpha\rangle + b|\beta\rangle) = \left(\frac{N-2}{N}a - \frac{2\sqrt{N-1}}{N}b\right)|\alpha\rangle + \left(\frac{2\sqrt{N-1}}{N}a + \frac{N-2}{N}b\right)|\beta\rangle. \tag{264}$$

Hence, the matrix for $\mathcal{G}$ in the basis $\{|\alpha\rangle, |\beta\rangle\}$ is

$$\mathcal{G} = \frac{1}{N}\begin{bmatrix} N-2 & -2\sqrt{N-1} \\ 2\sqrt{N-1} & N-2 \end{bmatrix}. \tag{265}$$

Therefore, one Grover iteration is an anti-clockwise rotation in the $|\alpha\rangle, |\beta\rangle$-plane by an angle of

$$\delta = \cos^{-1}\left(\frac{N-2}{N}\right) = \sin^{-1}\left(\frac{2\sqrt{N-1}}{N}\right). \tag{266}$$

7.1.2.1 Geometric picture This gives a simple geometric interpretation of Grover's algorithm. The entire evolution takes place inside the 2D subspace spanned by $|\alpha\rangle$ and $|\beta\rangle$, where $|\beta\rangle$ is the marked state and $|\alpha\rangle$ is the uniform superposition of all unmarked states. Although the Hilbert space has dimension N, Grover's algorithm effectively reduces the problem to repeated *planar* rotations. The initial state is

$$|+^n\rangle = \sqrt{\frac{N-1}{N}}|\alpha\rangle + \frac{1}{\sqrt{N}}|\beta\rangle. \tag{267}$$

Thus, we start very close to the $|\alpha\rangle$-axis, since the angle the initial vector makes with the $|\alpha\rangle$-axis is given by

$$\delta_0 = \tan^{-1}\left(\frac{1}{\sqrt{N-1}}\right) = \sin^{-1}\left(\frac{1}{\sqrt{N}}\right). \tag{268}$$

This allows us to express the initial state as

$$|+^n\rangle = \cos\delta_0|\alpha\rangle + \sin\delta_0|\beta\rangle. \tag{269}$$

Geometrically, the goal of Grover's algorithm is to rotate the state vector from near the $|\alpha\rangle$-axis toward the $|\beta\rangle$-axis. The oracle $\mathcal{O}$ reflects the state in the $|\alpha\rangle$-axis by negating the $|\beta\rangle$ component. The diffusion operator $2|+^n\rangle\langle+^n| - I_N$ then reflects in the line spanned by $|+^n\rangle$. Since a composition of two reflections is a rotation, one Grover iteration produces a rotation by angle δ. Note from (266) and (268) that

$$\delta = 2\delta_0. \tag{270}$$

Hence, after one Grover iteration, the state becomes

$$\begin{aligned} \mathcal{G}|+^n\rangle &= \cos(\delta_0+\delta)|\alpha\rangle + \sin(\delta_0+\delta)|\beta\rangle & (271)\\ &= \cos(3\delta_0)|\alpha\rangle + \sin(3\delta_0)|\beta\rangle. & (272)\end{aligned}$$

Thus, after t Grover iterations, the state becomes

$$\begin{aligned} \mathcal{G}^t|+^n\rangle &= \cos(\delta_0+t\delta)|\alpha\rangle + \sin(\delta_0+t\delta)|\beta\rangle & (273)\\ &= \cos((2t+1)\delta_0)|\alpha\rangle + \sin((2t+1)\delta_0)|\beta\rangle. & (274)\end{aligned}$$

Each iteration increases the amplitude on the marked state by rotating the vector closer to the $|\beta\rangle$-axis. The optimal stopping point occurs after $t_\star$ iterations when the state is as close as possible to $|\beta\rangle$; i.e., $(2t_\star+1)\delta_0 = \frac{\pi}{2}$. For large N,

$$\delta_0 = \sin^{-1}\left(\frac{1}{\sqrt{N}}\right) \approx \frac{1}{\sqrt{N}}. \tag{275}$$

Therefore,

$$t_\star \approx \frac{\pi}{4\delta_0} \approx \frac{\pi}{4}\sqrt{N}. \tag{276}$$

Thus, $\frac{\pi}{4}\sqrt{N}$ is the leading asymptotic term, not the exact value. The exact value for the number of Grover iterations is given by

$$t_\star = \left\lfloor \frac{\pi}{4\sin^{-1}\left(\frac{1}{\sqrt{N}}\right)} - \frac{1}{2} \right\rfloor. \tag{277}$$

7.1.3 Runtime complexity

The key point is that the algorithm amplifies the marked amplitude from $\frac{1}{\sqrt{N}}$ to almost 1 using only about $\frac{\pi}{4}\sqrt{N}$ oracle calls. This is a quadratic speedup: classical unstructured search needs $\Theta(N)$ queries, while Grover search needs only $\Theta\left(\sqrt{N}\right)$.

7.1.4 Case of multiple solutions

Suppose, instead of one solution, there are M possible solutions; i.e., we're looking for one of M things out of N. Then, we modify $\mathcal{O}$ such that it multiplies the input $|x\rangle$ by a phase of -1 if x is a solution.

$$|x\rangle \longrightarrow \boxed{\mathcal{O}} \longrightarrow (-1)^{\mathbb{I}(x \text{ is a solution})}|x\rangle. \tag{278}$$

$|\alpha\rangle$ and $|\beta\rangle$ are now defined to be:

$$|\alpha\rangle = \frac{1}{\sqrt{N-M}} \sum_{\substack{x \text{ not} \\ \text{marked}}} |x\rangle, \qquad |\beta\rangle = \frac{1}{\sqrt{M}} \sum_{x \text{ marked}} |x\rangle. \tag{279}$$

The quantum algorithm itself remains the same with $\mathcal{G} = (2|+^n\rangle\langle+^n| - I_N)\mathcal{O}$. We start with the state

$$|+\rangle^{\otimes n} = \sqrt{\frac{N-M}{N}}|\alpha\rangle + \sqrt{\frac{M}{N}}|\beta\rangle. \tag{280}$$

Hence, we start closer to the $|\alpha\rangle$-axis than in the $M = 1$ case with

$$\delta_0 = \sin^{-1}\left(\sqrt{\frac{M}{N}}\right), \qquad \delta = 2\delta_0. \tag{281}$$

Despite multiple solutions, we assume that $M \ll N$ since if the number of solutions is large as compared to N, we don't need a quantum algorithm at all: A randomly picked point is probably a solution. Hence,

$$t_\star = \left\lfloor \frac{\pi}{4\sin^{-1}\left(\sqrt{\frac{M}{N}}\right)} - \frac{1}{2} \right\rfloor \approx \left\lfloor \frac{\pi}{4}\sqrt{\frac{N}{M}} \right\rfloor \tag{282}$$

7.1.5 Case of unknown number of solutions

Suppose we don't know M *a priori*. A standard strategy is the Boyer-Brassard-Høyer-Tapp method. Instead of fixing the number of Grover iterations, we repeatedly choose a random iteration count from an expanding range. The algorithm proceeds roughly as follows:

1. Start with a small parameter $m = 1$.
2. Choose an integer $t \xleftarrow{\$} \{0, 1, \ldots, m-1\}$ uniformly at random.
3. Apply t Grover iterations to $|+\rangle^{\otimes n}$, then measure.
4. If a marked item is found, stop.
5. Otherwise, enlarge the search window: $m \leftarrow \lambda m$, for some constant $1 < \lambda < 2$ (often $\lambda = \frac{6}{5}$), and repeat.

7.1.5.1 Analysis For fixed m, the average success probability is

$$\overline{p}_m = \frac{1}{m} \sum_{t=0}^{m-1} \sin^2((2t+1)\delta_0) \tag{283}$$

$$= \frac{1}{2m} \sum_{t=0}^{m-1} 1 - \cos(2(2t+1)\delta_0). \tag{284}$$

Using the finite cosine-sum identity,

$$\overline{p}_m = \frac{1}{2} - \frac{\sin(4m\delta_0)}{4m\sin(2\delta_0)}. \tag{285}$$

Therefore,

$$\overline{p}_m \geq \frac{1}{2} - \frac{1}{4m\sin(2\delta_0)}. \tag{286}$$

So once $m \geq \frac{1}{\sin(2\delta_0)}$, we get $\overline{p}_m \geq \frac{1}{4}$. But

$$\sin(2\delta_0) = \sin\delta = \frac{2\sqrt{M(N-M)}}{N}, \tag{287}$$

so for $M \ll N$,

$$\frac{1}{\sin(2\delta_0)} \approx \frac{1}{2}\sqrt{\frac{N}{M}}. \tag{288}$$

Thus, once the window m reaches the Grover scale $\Theta\left(\sqrt{\frac{N}{M}}\right)$, a randomly chosen $t \sim \mathsf{Unif}(\{0, \ldots, m-1\})$ succeeds with constant probability. Before that point, the algorithm may fail often, but the cost is small because m grows geometrically:

$$1 + \lambda + \lambda^2 + \cdots + \Theta\left(\sqrt{\frac{N}{M}}\right) = O\left(\sqrt{\frac{N}{M}}\right). \tag{289}$$

After reaching that scale, each round succeeds with probability at least a constant, so only $O(1)$ more rounds are needed in expectation. Therefore the expected number of Grover iterations is

$$O\left(\sqrt{\frac{N}{M}}\right) \tag{290}$$

even without knowing M.

7.1.6 Optimality

Grover's algorithm is asymptotically optimal for unstructured quantum search. Bennett, Bernstein, Brassard, and Vazirani proved that any quantum algorithm that finds a marked item in an unstructured database of size N with bounded error must make at least $\Omega\left(\sqrt{N}\right)$ oracle queries. Thus Grover's algorithm, which succeeds using $\Theta\left(\sqrt{N}\right)$ queries, achieves the best possible asymptotic scaling. Intuitively, although quantum interference allows amplitudes of marked states to be amplified quadratically faster than classical probability accumulation, each oracle call can only change the state by a limited amount. Consequently, no quantum algorithm can rotate the state toward the marked subspace substantially faster than the Grover scale. This establishes Grover search as the optimal quantum algorithm for black-box unstructured search.