

In memory of Nishant (Dec 2, 1994 - Apr 10, 2022), who was deeply passionate about cryptography.

CS 598 CTO: Quantum Cryptography

Lecturer: Dakshita Khurana

TA and Editor: Amit Agarwal

Prologue

The extended Church–Turing thesis asserts that any physically realizable model of computation can be efficiently simulated (i.e. with polynomial overhead) on a standard computational model such as a Turing machine, a Boolean circuit, a random access machine, or a cellular automaton. This thesis underpins classical complexity theory; for instance, it ensures that the class P (polynomial time) is well-defined. The contrapositive states that any task lying outside P (or BPP, if randomization is permitted) cannot be performed by any physical device.

Consider the following problem: given $N = pq$, where p and q are randomly sampled primes, recover p and q . This task is widely believed to be intractable for classical polynomial-sized circuits and is therefore conjectured to lie outside P. However, as we will see in Lecture 7, there exists a *quantum* polynomial-sized circuit that factors products of large primes. Thus, if large-scale quantum computers were physically realizable, they would refute the extended Church–Turing thesis.¹

In the first part of this course, we develop the tools and concepts required to understand the exponential speedups that quantum circuits provide. A common misconception is that quantum computers can efficiently² solve *all* hard problems, including NP-complete ones. This is unlikely to be true. While a quantum algorithm can in principle place all candidate solutions to a problem into superposition, extracting useful information requires concentrating a large probability mass on the correct solution. Shor demonstrated how this can be done for highly structured problems such as factoring, but these techniques are not believed to extend to NP-hard problems in general.

The advent of efficient quantum factoring will have profound consequences for much of the cryptography in use today. In the second part of the course, we examine how to build cryptosystems that remain secure against quantum attacks. The central technique is a *reduction*: any adversary that breaks an advanced protocol is transformed into one that violates a basic mathematical assumption believed to evade quantum attacks. We will focus on the key ideas from recent cryptographic research and identify promising directions for the future.

Finally, we will see surprising cryptographic tasks that become achievable when one or more participants have access to a quantum computer. A classical string can be copied arbitrarily many times, and an adversary holding it can never be compelled to delete it. Quantum information, by contrast, is unclonable, and measurement can be used to destroy it. These properties make quantum information uniquely well-suited to systems that exploit forced “deletion” or the inability of an adversary to duplicate an unknown state. We will study several such applications in the final part of the course.

¹An “amended” thesis stating that physically realizable models of computation can be efficiently simulated by a *quantum* circuit may nonetheless still hold.

²In this course, “efficiently” will always mean “in time polynomial in the input length.”

Contents

1	Reversible Computation and Complex Numbers	9
1.1	Reversible Computation	9
1.2	Complex Numbers	10
2	Quantum Mechanics: Model, Unitaries, Measurement	13
2.1	Origins: Young's Double-Slit Experiment	13
2.2	Quantum States	13
2.2.1	Representing a Quantum State	13
2.3	Measurement	15
2.4	State Evolution: Qubits and Quantum Operations	16
2.4.1	Phase shifts	16
2.4.2	Bit flip	16
2.4.3	Hadamard transform	17
2.4.4	Unitary evolution	17
2.5	Multi-Qubit Systems	18
2.5.1	Measuring multi-qubit states	18
2.6	Further Reading	19
3	Quantum Gates and Circuits; Entanglement; No-Cloning	20
3.1	Inner and Outer Products	20
3.1.1	Inner Product	20
3.1.2	Outer Product	21
3.2	Quantum Circuits	21
3.2.1	2-qubit unitary gates: example	22
3.2.2	3-qubit unitary gates: example	22
3.2.3	Building Multi-Gate Circuits	22
3.2.4	Entanglement	23
3.2.5	No-Cloning	23
3.3	Further Reading	24
4	Quantum Algorithms	25
4.1	The Toffoli (CCNOT) Gate	25
4.2	Reversible Circuits and Uncomputing	25
4.2.1	Example: Reversible Circuit for a NAND-NAND Composition	25
4.2.2	Uncomputing	26
4.3	The Deutsch-Jozsa Algorithm	26
4.3.1	Quantum Implementation of Classical Functions	27
4.3.2	The Phase-Kickback Trick	27
4.3.3	Warm-up: $n = 1$	27
4.3.4	The General Algorithm	28
4.4	Further Reading	28

5	Quantum Fourier Transform	29
5.1	The Deutsch-Jozsa Algorithm	29
5.2	Fourier Transform over \mathbb{F}_2^n	30
5.3	Simon's Algorithm	31
5.4	Further Reading	32
6	Fourier Basis, Simon's Algorithm, and Period Finding	33
6.1	Functions as Quantum States	33
6.1.1	Functions in the Fourier Basis	33
6.2	Revisiting Deutsch-Jozsa via Fourier Analysis	34
6.3	Simon's Algorithm Revisited	35
6.4	Fourier Transform over \mathbb{Z}_N	35
6.4.1	Geometry of ω_N	35
6.4.2	Properties of χ_s	36
6.5	Period Finding (Preview)	36
6.6	Further Reading	37
7	Period-finding and Shor's Algorithm	38
7.1	Period-Finding	38
7.2	Shor's Algorithm	40
7.3	The Hidden Subgroup Problem	41
7.4	Implementing QFT over \mathbb{Z}_N	41
7.5	Further Reading	42
8	Implementation of the QFT and Introduction to Mixed States	43
8.1	QFT for $N = 16$	43
8.2	Applications of the QFT	45
8.3	Mixed States	45
8.4	Further Reading	46
9	Properties of Mixed States and Introduction to Quantum Key Distribution	47
9.1	Measuring Pure and Mixed States	47
9.2	Properties of Density Matrices	47
9.3	Unitary Operations on Mixed States	48
9.4	Spectral Form of a Density Matrix	49
9.5	The Maximally Mixed State	49
9.6	Quantum Key Distribution: Setup	50
9.7	Further Reading	51
10	Quantum Key Distribution	52
10.1	Problem Setup	52
10.2	Indistinguishability	52
10.3	Attempt 0: Classical Channel Only	53
10.4	Attempt 1: Encoding in Random Bases	53
10.5	Attempt 2: Test Step	54
10.6	An Alternative Perspective: EPR Pairs	55

10.7	The Adversarial View	56
10.8	Sampling Game	56
10.9	Putting It Together	58
11	Quantum Key Distribution – II	59
11.1	Agreement: Intuition	59
11.2	Secrecy: Intuition	60
11.3	Classical Sampling and Estimation	60
11.3.1	Step 1: Relating $w[q_T]$ and $w[q_S]$	60
11.3.2	Step 2: Relating $w[q_T]$ and $w[q_{\bar{T}}]$	61
11.3.3	Step 3: Combining Steps 1 and 2	61
11.3.4	Converting Classical to Quantum Bounds	61
11.4	Further Reading	62
12	Finishing QKD and Introduction to Quantum Oblivious Transfer	63
12.1	Error Correction in QKD	63
12.2	Privacy in QKD	64
12.3	Oblivious Transfer	65
12.4	Security Against Malicious Senders	65
12.5	Security Against Malicious Receivers	66
12.6	Attempt: Information-Theoretically Secure Quantum OT	67
12.7	Commitment Schemes	68
13	Quantum Oblivious Transfer – II	69
13.1	Commitment Schemes	69
13.1.1	Indistinguishability	69
13.2	Quantum Oblivious Transfer	70
13.2.1	Proof of Measurement	70
13.2.2	Security Against Malicious Bob: Setup	71
14	Quantum Oblivious Transfer – III	73
14.1	Security Against Malicious Bob	73
14.1.1	The Variant QOT*	73
14.1.2	Security of QOT*	74
14.2	Applications of OT	75
14.3	Further Reading	76
15	Quantum Random Oracles, Introduction to Encryption	77
15.1	The Random Oracle Model	77
15.2	Symmetric Encryption	78
15.3	Notions of Security	78
15.3.1	Information-Theoretic Security	78
15.3.2	One-Time Pad	78
15.3.3	CPA Security: Single Message	79
15.3.4	CPA Security: Multi-Message	80
15.4	Learning With Errors	80

15.4.1	Search-LWE	81
15.4.2	Decisional LWE	81
16	Encrypting Classical Bits and Quantum States	82
16.1	Multi-Message Security: Recap	82
16.1.1	The Adversary–Challenger Game	82
16.1.2	Why the One-Time Pad Fails for Multiple Messages	83
16.2	Multi-Message Encryption from LWE	83
16.2.1	Reviewing the LWE Assumption	83
16.2.2	Single-Message Encryption from LWE	83
16.2.3	Multi-Message Encryption	84
16.3	Quantum One-Time Pad	85
16.4	Combining the QOTP with LWE Encryption	86
17	Private-Key Encryption for Classical Bits and Qubits	87
17.1	The Quantum One-Time Pad and Pauli Gates	87
17.1.1	The Pauli X Gate	87
17.1.2	The Pauli Z Gate	87
17.1.3	The Quantum One-Time Pad in Full Generality	88
17.1.4	Geometric Interpretation: the Bloch Sphere	88
17.2	Private-Key Encryption	88
17.2.1	One-Time vs. Multi-Message Security: Recap	89
17.2.2	Encryption of Qubits	89
17.2.3	Correctness	90
17.2.4	Security	90
18	Public-Key Encryption for Classical Bits and Qubits from LWE	91
18.1	Definition of Public-Key Encryption	91
18.2	Single-Message vs. Multi-Message Security	91
18.2.1	The Limited Multi-Message Adversary	92
18.2.2	The Hybrid Argument	92
18.3	Regev Encryption	93
18.3.1	Correctness	93
18.3.2	Security	94
18.4	Public-Key Encryption of Qubits	94
18.4.1	Correctness	95
18.4.2	Security	95
19	Homomorphic Encryption for Classical Circuits	96
19.1	Regev Public-Key Encryption Recap	96
19.2	The GSW FHE Scheme	97
20	Homomorphic Encryption for Quantum Circuits	100
20.1	Quantum Homomorphic Encryption: Definition	100
20.2	A Scheme for Clifford Circuits	100
20.2.1	Setup	100

20.2.2	Homomorphic Evaluation: Single-Qubit Cliffords	101
20.2.3	Two-Qubit Cliffords: CNOT	101
20.2.4	Limitation: Non-Clifford Gates	101
20.3	Encrypted CNOT	102
20.4	Trapdoor Claw-Free Functions	102
20.5	The Plan for Quantum FHE	103
20.6	Further Reading	103
21	Quantum Fully Homomorphic Encryption – II	104
21.1	The Toffoli Gate	104
21.2	Encrypted CNOT: Setup	104
21.3	From FHE to TCFs	105
21.3.1	TCF Construction Recap	105
21.4	Entangling a Claw With $ \psi\rangle$	105
21.4.1	Splitting the Claw	106
21.4.2	Removing the Residue via QFT	106
21.4.3	Reading Off the Pauli Mask	106
21.5	Putting It All Together	107
21.6	Further Reading	107
22	Homomorphic Encryption III	108
22.1	Step 1: TCF Encryption	108
22.2	Step 2: Implementing Encrypted CNOT from TCF-Enc(s)	109
22.3	Implementing Step 1: From HE-Enc(s) to TCF-Enc(s)	110
22.4	Applications of Quantum FHE	111
22.5	References	111
23	Testing A Qubit	112
23.1	Motivation	112
23.2	The Construction at a High Level	112
23.3	Detailed Protocol	113
23.4	Why This Tests a Qubit	114
23.5	Cryptographic Test of Quantumness	114
23.6	Further Reading	115
24	Testing a Qubit – II; Circuit-to-Hamiltonian	116
24.1	Recap and Security Statement	116
24.2	Security Argument	116
24.3	Circuit-to-Hamiltonian Reduction	117
24.3.1	Setup	117
24.3.2	The Hamiltonian	118
24.3.3	Why This Helps	118
24.4	Further Reading	118

25 Classical Verification of Quantum Computation	119
25.1 Local Hamiltonian	119
25.2 The Fitzsimons–Morimae Protocol	120
25.3 Further Reading	122
25.4 Quantum Money	122
25.5 Model	122
25.6 Verification	123
25.7 f_k is a Pseudorandom Function (PRF)	123
25.8 Security Intuition	125
26 Quantum Money (Continued); the Zeno Effect; Quantum Lightning; Cryptography with Deletion	126
26.1 Recap from Last Lecture	126
26.2 The Quantum Zeno Effect	126
26.3 The Elitzur–Vaidman Bomb Tester Problem	128
26.4 An Adaptive Attack on Wiesner’s Quantum Money	129
26.5 Quantum Lightning	132
26.6 Cryptography with Deletion	132

1 Reversible Computation and Complex Numbers

As preparation for defining the quantum circuit model, we first review reversible classical computation and the algebra of complex numbers.

1.1 Reversible Computation

Definition 1. A Boolean gate G is *reversible* if it has the same number of inputs as outputs and its input-to-output mapping is a bijection.

The NOT gate is reversible. By contrast, classical gates such as AND, OR, XOR, and NAND are not reversible: they have more inputs than outputs, and their outputs lose information about at least some inputs.

We begin by describing a reversible analogue of the XOR gate, called the *controlled*-NOT (or CNOT) gate, depicted in Figure 1.

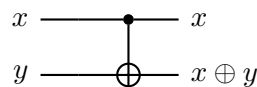


Figure 1: The CNOT gate.

The truth table is given in Table 2. Equivalently, the CNOT gate flips its second bit if and only if the first bit is 1: when the *control* bit $x = 0$, the second bit y is unchanged, while when $x = 1$, y is replaced by $\text{NOT}(y)$.

Input	Output
00	00
01	01
10	11
11	10

Figure 2: Truth table of the CNOT gate.

The notion of a controlled gate extends to any gate. Given an arbitrary gate G with n input and output wires, the *controlled- G* gate acts on $n+1$ wires: a control wire x , and n data wires on which G is applied if and only if $x = 1$. Figure 3 illustrates the case $n = 3$. The output is $xc_1c_2c_3 = xa_1a_2a_3$ when $x = 0$, and $xc_1c_2c_3 = xb_1b_2b_3$ when $x = 1$, where G maps $a_1a_2a_3 \mapsto b_1b_2b_3$.

Specializing to $G = \text{CNOT}$ yields the controlled-controlled-NOT (or CCNOT) gate, shown in Figure 11. When $x = 1$, a CNOT is applied to (y, z) ; when $x = 0$, both wires are left untouched.

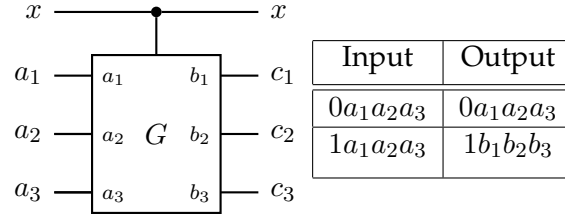


Figure 3: The controlled- G gate and its truth table.

Since CNOT leaves its control bit y unchanged in either case, y is always preserved. The third bit z is replaced by $z \oplus y$ when $x = 1$ and by z when $x = 0$. Writing this as a single Boolean formula, z is replaced by

$$(x \wedge (z \oplus y)) \oplus ((\neg x) \wedge z), \quad (1.1)$$

since when $x = 0$ only the second term survives (yielding z), and when $x = 1$ only the first survives (yielding $z \oplus y$). Equation (1.1) simplifies as follows:

$$\begin{aligned} (x \wedge (z \oplus y)) \oplus ((\neg x) \wedge z) &= (x \wedge z) \oplus (x \wedge y) \oplus ((\neg x) \wedge z) \\ &= ((x \oplus \neg x) \wedge z) \oplus (x \wedge y) \\ &= z \oplus (x \wedge y). \end{aligned}$$

Thus the CCNOT gate preserves the first two bits and replaces the third with $z \oplus (x \wedge y)$ — a reversible implementation of the AND gate.

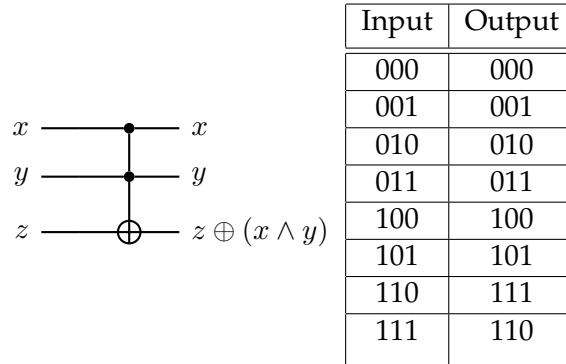


Figure 4: The CCNOT (Toffoli) gate and its truth table.

1.2 Complex Numbers

We briefly recall the algebra of complex numbers. A *complex number* has the form $z = a + bi$ for $a, b \in \mathbb{R}$, where $i = \sqrt{-1}$ is the imaginary unit. We call a the *real part* and b the *imaginary part* of z . The *magnitude* of z is

$$|z| = \sqrt{a^2 + b^2}.$$

We may also express z in polar form:

$$z = |z| \cos \theta + i|z| \sin \theta = |z|e^{i\theta},$$

where $|z| = \sqrt{a^2 + b^2}$ is the *radial coordinate* and $\theta = \cos^{-1}\left(\frac{a}{\sqrt{a^2+b^2}}\right)$ is the *angular coordinate*.

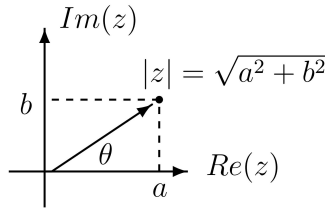


Figure 5: Geometric representation of $z = a + bi$, image courtesy [OW15].

Complex conjugate. The *complex conjugate* of z is denoted z^* (or z^\dagger , pronounced “zee-dagger”). For $z = a + bi$, we have $z^* = a - bi$. Note that $|z^*| = \sqrt{a^2 + (-b)^2} = |z|$.

Products of complex numbers. For $z_1 = a_1 + b_1i$ and $z_2 = a_2 + b_2i$,

$$\begin{aligned} z_1 \cdot z_2 &= (a_1 + b_1i)(a_2 + b_2i) \\ &= a_1a_2 + b_1b_2i^2 + a_1b_2i + a_2b_1i \\ &= (a_1a_2 - b_1b_2) + (a_1b_2 + a_2b_1)i. \end{aligned}$$

In polar form, multiplication is particularly simple: if $z_1 = |z_1|e^{i\theta_1}$ and $z_2 = |z_2|e^{i\theta_2}$, then

$$z_1 \cdot z_2 = |z_1||z_2|e^{i(\theta_1+\theta_2)}.$$

The product of $z = a + bi$ with its conjugate gives the squared magnitude:

$$zz^* = (a + bi)(a - bi) = a^2 - b^2i^2 = a^2 + b^2 = |z|^2.$$

Conjugate transpose of a vector. An $m \times n$ *complex matrix* is an array of complex numbers with m rows and n columns. A complex *vector* is an $m \times 1$ matrix. The conjugate transpose A^\dagger of a complex column vector

$$A = \begin{bmatrix} \alpha \\ \beta \\ \vdots \\ \eta \end{bmatrix}$$

is the row vector

$$A^\dagger = [\alpha^* \ \beta^* \ \cdots \ \eta^*].$$

Two vectors A and B are *orthogonal* if $A^\dagger B = 0$. *Orthonormality* additionally requires $A^\dagger A = B^\dagger B = 1$.

Conjugate transpose of a matrix. For an $m \times n$ complex matrix M , the conjugate transpose M^\dagger is the $n \times m$ matrix obtained by first transposing M and then taking the complex conjugate of each entry.

Key Takeaways. Reversible gates have equal numbers of inputs and outputs and act bijectively; the CCNOT gate is a reversible implementation of AND. Complex numbers admit both Cartesian and polar representations, the latter making multiplication a sum of phases, a fact we will exploit throughout the course.

2 Quantum Mechanics: Model, Unitaries, Measurement

To understand the (computational, pun intended) basis on which quantum computing rests, we begin with a short introduction to the phenomena of quantum mechanics that make this model of computation possible.

2.1 Origins: Young's Double-Slit Experiment

The double-slit experiment, originally due to Thomas Young, makes the wave–particle duality of light explicit by splitting a beam of light and observing the resulting pattern on a screen. Young's original setup passed sunlight through a single hole, but modern variants use a coherent single-photon source and two slits; the result is an interference pattern characteristic of waves (left panel of Figure 6). Identical patterns arise with, e.g., water waves, providing direct evidence of the wave nature of light.

If one adds a detector that observes which slit each photon passes through, the wave behavior disappears: each photon is now seen to traverse exactly one slit, and the screen shows two distinct bands corresponding to the particle nature of light (right panel of Figure 6). One says that observation *collapses the wave function* of the photon. Without the detector, the photon exists in a *superposition* of the two states “top slit” and “bottom slit,” effectively passing through both at once.

Quantum computing exploits precisely this notion of superposition. Going forward, we abandon English descriptions in favor of a more compact notation: the state “the photon takes the top slit” is written $|0\rangle$ (read “ket zero”), and “the photon takes the bottom slit” is written $|1\rangle$ (read “ket one”).

2.2 Quantum States

As Section 2.1 suggests, a quantum state can be in a *superposition* of two basis states. We record this as a theorem.

Theorem 2.1. *If a quantum system can be in state $|0\rangle$ or state $|1\rangle$, it can also be in any superposition*

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle,$$

where $\alpha, \beta \in \mathbb{C}$ are amplitudes satisfying the normalization condition $|\alpha|^2 + |\beta|^2 = 1$. The probability of observing the system in state $|0\rangle$ upon measurement is $|\alpha|^2$, and the probability of observing $|1\rangle$ is $|\beta|^2$. Every two-state quantum system can be written in this way as a linear combination of basis states.

2.2.1 Representing a Quantum State

The notation introduced above is called *Dirac notation* (or *bra-ket notation*). Several examples illustrate it.

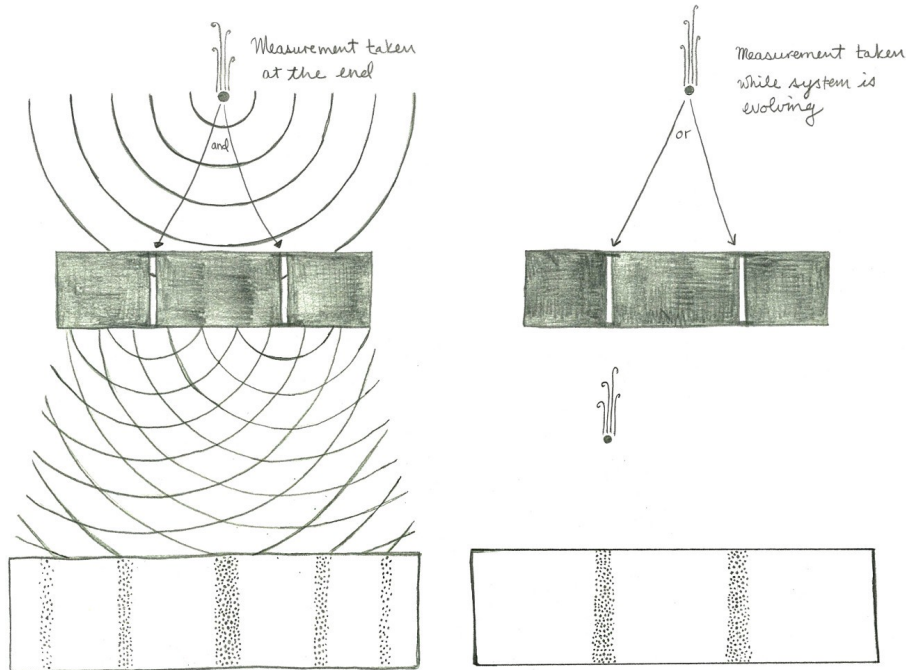


Figure 6: A double-slit experiment. (Left) Without observation, an interference pattern emerges, evidencing the wave nature of light. (Right) Observing the slits collapses the photon's wave function, and the screen shows two bands corresponding to particles passing through one or the other slit. Image: Medium.

Example 2.1. Take $\alpha = 0.8, \beta = 0.6$, so that $|\Psi\rangle = 0.8|0\rangle + 0.6|1\rangle$. Then $\Pr[\text{observe } |0\rangle] = 0.64$ and $\Pr[\text{observe } |1\rangle] = 0.36$. The normalization $|\alpha|^2 + |\beta|^2 = 1$ holds.

Example 2.2. The state $|\chi\rangle = 0.8|0\rangle - 0.6|1\rangle$ has the same measurement probabilities as $|\Psi\rangle$ above but is a *distinct* state, since its amplitudes differ.

Example 2.3. Amplitudes may be complex. The state $|\Omega\rangle = i|0\rangle + 0 \cdot |1\rangle$ satisfies $|i|^2 + 0^2 = 1$ and yields outcome $|0\rangle$ with certainty upon measurement.

Complex amplitudes, as in Example 2.3, let states evolve continuously, a feature we will exploit later in the course. For the moment we restrict attention to real amplitudes.

The states above are *kets*. Their *conjugate transposes* (or *daggers*), called *bras*, will be equally useful. To convert a ket $|\Psi\rangle$ into the bra $\langle\Psi|$, flip the bracket and replace each amplitude with its complex conjugate.

Theorem 2.2. The conjugate transpose of $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is

$$\langle\Psi| = (|\Psi\rangle)^\dagger = \alpha^* \langle 0| + \beta^* \langle 1|.$$

Equivalently, every quantum state admits a matrix representation: *kets are column vectors and bras are row vectors*. The conjugate transpose terminology now reflects an actual matrix operation.

Example 2.4. The state $|\Psi\rangle = 0.8|0\rangle + 0.6|1\rangle$ of Example 2.1 corresponds to the column vector $\begin{bmatrix} 0.8 \\ 0.6 \end{bmatrix}$.

Example 2.5. The state $|\chi\rangle = 0.8|0\rangle - 0.6|1\rangle$ corresponds to $\begin{bmatrix} 0.8 \\ -0.6 \end{bmatrix}$.

Example 2.6. The state $|\Omega\rangle = i|0\rangle + 0 \cdot |1\rangle$ corresponds to $\begin{bmatrix} i \\ 0 \end{bmatrix}$.

Theorem 2.3. In general, $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$ corresponds to the column vector $\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$ with complex entries, and its conjugate transpose $\langle\Psi| = \alpha^*\langle 0| + \beta^*\langle 1|$ corresponds to the row vector $[\alpha^* \ \beta^*]$. Any such column vector can be expanded in the orthonormal computational basis

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

since $\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \alpha \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \beta \begin{bmatrix} 0 \\ 1 \end{bmatrix}$.

Theorem 2.4. (Inner product.) For any ket $|\Psi\rangle$, the inner product with its bra is $\langle\Psi|\Psi\rangle = 1$. In matrix notation,

$$[\alpha^* \ \beta^*] \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \alpha^*\alpha + \beta^*\beta = |\alpha|^2 + |\beta|^2 = 1.$$

So far we have expressed quantum states in the *computational basis* $\{|0\rangle, |1\rangle\}$. There are infinitely many other orthonormal bases. Another common choice is the *Hadamard basis* $\{|+\rangle, |-\rangle\}$, defined by

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

To see how a state changes representation, expand $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$ in the (non-normalized) basis $\left\{ \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \end{bmatrix} \right\}$ with new amplitudes α', β' :

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \alpha' \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \beta' \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Solving gives $\alpha' = \frac{\alpha+\beta}{2}$ and $\beta' = \frac{\alpha-\beta}{2}$. Normalizing the basis vectors, we obtain

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \frac{\alpha+\beta}{\sqrt{2}}|+\rangle + \frac{\alpha-\beta}{\sqrt{2}}|-\rangle.$$

2.3 Measurement

We now formalize what we have been calling “measurement.” A measurement collapses a quantum state into one of the basis states of the chosen measurement basis, with probabilities given by the squared amplitudes. In a circuit diagram, a measurement is depicted as in Figure 7.

Measuring $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$ in the computational basis yields outcome “0” with probability $|\alpha|^2$ and outcome “1” with probability $|\beta|^2$. Once a wave function has collapsed, subsequent measurements in the same basis return the same outcome with certainty, since the state is no longer in superposition. Figure 8 illustrates this for outcome “0.”

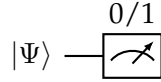


Figure 7: A measurement of $|\Psi\rangle$ yields “0” or “1.”

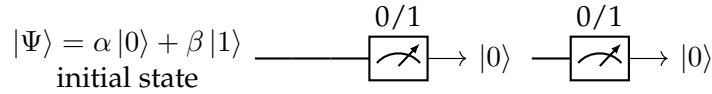


Figure 8: With probability $|\alpha|^2$, the first measurement outputs “0,” leaving the post-measurement state $|0\rangle$. Subsequent computational-basis measurements then always output “0.”

2.4 State Evolution: Qubits and Quantum Operations

For the purposes of computing, the two-level quantum systems above are called *qubits*, the quantum analogue of bits. We now turn to operations on a single qubit.

2.4.1 Phase shifts

Returning to the double-slit experiment of Section 2.1, changing the medium in which one slit is carved can introduce a *phase shift* in the interference pattern. In Dirac notation, a phase shift on basis state $|1\rangle$ multiplies its amplitude by $e^{i\theta}$:

$$\frac{|0\rangle+|1\rangle}{\sqrt{2}} \longrightarrow \frac{|0\rangle+e^{i\theta}|1\rangle}{\sqrt{2}}.$$

In matrix form this is

$$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \alpha \\ e^{i\theta}\beta \end{bmatrix}.$$

A phase shift on $|0\rangle$ instead is described by

$$\begin{bmatrix} e^{i\theta} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} e^{i\theta}\alpha \\ \beta \end{bmatrix}.$$

Example 2.7. For $\theta = \pi$, $e^{i\theta} = -1$, so $\alpha|0\rangle + \beta|1\rangle \mapsto \alpha|0\rangle - \beta|1\rangle$.

Example 2.8. For $\theta = \frac{\pi}{2}$, $e^{i\theta} = i$, so $\alpha|0\rangle + \beta|1\rangle \mapsto \alpha|0\rangle + i\beta|1\rangle$.

2.4.2 Bit flip

A bit flip exchanges $|0\rangle \leftrightarrow |1\rangle$. The corresponding matrix is the *Pauli-X* gate (a quantum NOT),

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad X \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \beta \\ \alpha \end{bmatrix}.$$

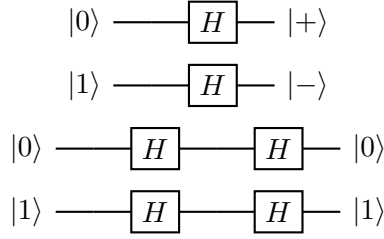


Figure 9: The Hadamard gate changes basis. Two applications return the original state.

2.4.3 Hadamard transform

The *Hadamard transform* converts the computational basis into the Hadamard basis, and vice versa (Figure 9).

In matrix form,

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

Two applications return the identity:

$$HH = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} = \mathbb{I}.$$

2.4.4 Unitary evolution

A general single-qubit transformation U acts linearly on each basis state:

$$\begin{aligned} |0\rangle &\longrightarrow U_{00}|0\rangle + U_{01}|1\rangle, \\ |1\rangle &\longrightarrow U_{10}|0\rangle + U_{11}|1\rangle, \end{aligned}$$

i.e. as the matrix³

$$U = \begin{bmatrix} U_{00} & U_{10} \\ U_{01} & U_{11} \end{bmatrix}.$$

For an input qubit $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$, the output state is

$$U|\Psi\rangle = \begin{bmatrix} U_{00} & U_{10} \\ U_{01} & U_{11} \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} U_{00}\alpha + U_{10}\beta \\ U_{01}\alpha + U_{11}\beta \end{bmatrix} = |\Phi\rangle.$$

For U to describe a valid quantum operation, $|\Phi\rangle$ must itself be a unit vector, i.e. $\langle\Phi|\Phi\rangle = 1$. Recalling that the dagger reverses the order of products, $(AB)^\dagger = B^\dagger A^\dagger$, we compute

$$\langle\Phi|\Phi\rangle = (U|\Psi\rangle)^\dagger(U|\Psi\rangle) = \langle\Psi|U^\dagger U|\Psi\rangle = 1.$$

Since this holds for every unit vector $|\Psi\rangle$, we conclude $U^\dagger U = \mathbb{I}$.

³Note that we follow the standard convention U_{ij} denotes the entry in row i , column j when U is written as $U = \begin{bmatrix} U_{00} & U_{10} \\ U_{01} & U_{11} \end{bmatrix}$; this matches how column vectors transform.

Definition 2. A matrix U satisfying $U^\dagger U = \mathbb{I}$ is called *unitary*. Equivalently:

1. $U^\dagger = U^{-1}$, i.e. the conjugate transpose of U is its inverse.
2. Writing out the matrix product, $(U^\dagger U)_{ij} = \sum_k (U^\dagger)_{ik} U_{kj} = \sum_k U_{ki}^* U_{kj}$; this equals 1 if $i = j$ and 0 otherwise.

Definition 3. A matrix U is *Hermitian* (or *self-adjoint*) if $U = U^\dagger$.

The key point is that *every unitary matrix corresponds to a valid quantum operation on a qubit*.

2.5 Multi-Qubit Systems

So far we have considered single-qubit systems and 2×2 unitaries. Dirac notation extends naturally to multiple qubits. The *two-qubit computational basis* has four elements:

$$|00\rangle \equiv |0\rangle \otimes |0\rangle, \quad |01\rangle \equiv |0\rangle \otimes |1\rangle, \quad |10\rangle \equiv |1\rangle \otimes |0\rangle, \quad |11\rangle \equiv |1\rangle \otimes |1\rangle,$$

where \otimes denotes the *tensor product*.

If the two qubits are in (separate) states $|\Phi_0\rangle = \alpha_0 |0\rangle + \beta_0 |1\rangle$ and $|\Phi_1\rangle = \alpha_1 |0\rangle + \beta_1 |1\rangle$, the joint state is

$$\begin{aligned} |\Psi\rangle &= |\Phi_0\rangle \otimes |\Phi_1\rangle \\ &= (\alpha_0 |0\rangle + \beta_0 |1\rangle) \otimes (\alpha_1 |0\rangle + \beta_1 |1\rangle) \\ &= \alpha_0 \alpha_1 |00\rangle + \alpha_0 \beta_1 |01\rangle + \beta_0 \alpha_1 |10\rangle + \beta_0 \beta_1 |11\rangle. \end{aligned}$$

That this is a valid quantum state follows from

$$\begin{aligned} &|\alpha_0 \alpha_1|^2 + |\alpha_0 \beta_1|^2 + |\beta_0 \alpha_1|^2 + |\beta_0 \beta_1|^2 \\ &= |\alpha_0|^2 (|\alpha_1|^2 + |\beta_1|^2) + |\beta_0|^2 (|\alpha_1|^2 + |\beta_1|^2) \\ &= |\alpha_0|^2 + |\beta_0|^2 = 1. \end{aligned}$$

Relabeling, a general two-qubit state has the form

$$|\Psi\rangle = \alpha_{00} |00\rangle + \alpha_{01} |01\rangle + \alpha_{10} |10\rangle + \alpha_{11} |11\rangle, \quad \sum_{x \in \{0,1\}^2} |\alpha_x|^2 = 1.$$

2.5.1 Measuring multi-qubit states

We may measure *either* qubit alone, or both simultaneously. If only the first qubit is measured (in the computational basis), the probability of outcome $|0\rangle$ is the sum of squared amplitudes for those basis states with $|0\rangle$ in the first position:

$$\Pr[\text{first qubit} = |0\rangle] = |\alpha_{00}|^2 + |\alpha_{01}|^2.$$

The post-measurement state is the projection onto the relevant subspace, renormalized:

$$|\Psi'\rangle = |0\rangle \otimes \frac{\alpha_{00} |0\rangle + \alpha_{01} |1\rangle}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}} = \frac{\alpha_{00} |00\rangle + \alpha_{01} |01\rangle}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}}.$$

The denominator is the renormalization factor required so that the post-measurement state has unit norm.

Key Takeaways. A qubit is a unit vector $\alpha|0\rangle + \beta|1\rangle \in \mathbb{C}^2$, and a single-qubit operation is a unitary matrix U , characterized by $U^\dagger U = \mathbb{I}$. Important examples include the Pauli X (bit flip), phase shifts, and the Hadamard gate, which switches between the computational and Hadamard bases. Multi-qubit systems live in tensor-product spaces and admit partial measurements that collapse only some qubits, renormalizing the remaining amplitudes.

2.6 Further Reading

The standard reference for this material is Nielsen and Chuang [NC16]. In the 10th-anniversary edition: qubits (S1.2 and S2.2.1); gates, operations, and evolution (S1.3, S1.4, S2.2.2); the Hadamard and computational bases (S1.3.3); measurement (S2.2.3).

3 Quantum Gates and Circuits; Entanglement; No-Cloning

In the previous lecture we discussed pure two-qubit states. We now generalize: an n -qubit quantum state is a unit vector in \mathbb{C}^{2^n} . To work with such states it helps to fix a basis. Recall that for $n = 1$ the standard basis is $\{|0\rangle, |1\rangle\}$, and for $n = 2$ it is $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$:

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

$$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad |01\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad |10\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad |11\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

More generally, an n -qubit basis consists of the 2^n states $\{|x\rangle\}_{x \in \{0,1\}^n}$, corresponding to the corners of the n -dimensional Boolean hypercube. An n -qubit state may then be written as

$$|\Psi\rangle = \sum_{x \in \{0,1\}^n} \alpha_x |x\rangle, \quad \alpha_x \in \mathbb{C}, \quad \sum_x |\alpha_x|^2 = 1.$$

3.1 Inner and Outer Products

3.1.1 Inner Product

For states $|\Psi\rangle = \sum_x \alpha_x |x\rangle$ and $|\phi\rangle = \sum_x \beta_x |x\rangle$, the *inner product* on this complex Hilbert space is

$$\langle \Psi | \phi \rangle = \sum_{x \in \{0,1\}^n} \alpha_x^* \beta_x.$$

Equivalently, in matrix form, writing

$$|\Psi\rangle = \begin{bmatrix} \alpha_{0\dots 0} \\ \vdots \\ \alpha_{1\dots 1} \end{bmatrix}, \quad |\phi\rangle = \begin{bmatrix} \beta_{0\dots 0} \\ \vdots \\ \beta_{1\dots 1} \end{bmatrix},$$

the bra is the conjugate transpose, $\langle \Psi | = [\alpha_{0\dots 0}^* \ \cdots \ \alpha_{1\dots 1}^*]$, and

$$\langle \Psi | \phi \rangle = \langle \Psi | \phi \rangle = \sum_x \alpha_x^* \beta_x.$$

Remark. If $\langle \Psi | \phi \rangle = 0$, the states are *orthogonal*. In general $\langle \Psi | \phi \rangle \in \mathbb{C}$, but $\langle \Psi | \Psi \rangle \in \mathbb{R}$, and any valid quantum state satisfies $\langle \Psi | \Psi \rangle = 1$.

3.1.2 Outer Product

The *outer product* of $|\phi\rangle$ and $\langle\Psi|$ is the $2^n \times 2^n$ matrix

$$\begin{aligned} |\phi\rangle\langle\Psi| &= \begin{bmatrix} \beta_{0\dots 0} \\ \vdots \\ \beta_{1\dots 1} \end{bmatrix} [\alpha_{0\dots 0}^* \quad \dots \quad \alpha_{1\dots 1}^*] \\ &= \sum_{x,y \in \{0,1\}^n} \beta_x \alpha_y^* |x\rangle\langle y|, \end{aligned}$$

whose diagonal sum recovers the inner product,

$$\text{tr}(|\phi\rangle\langle\Psi|) = \sum_x \alpha_x^* \beta_x = \langle\Psi|\phi\rangle.$$

Remark. Useful trace identities for square matrices A, B of compatible size and $z \in \mathbb{C}$:

$$\begin{aligned} \text{tr}(AB) &= \text{tr}(BA), \\ \text{tr}(A+B) &= \text{tr}(A) + \text{tr}(B), \\ \text{tr}(zA) &= z\text{tr}(A), \\ \overline{\text{tr}(A)} &= \text{tr}(A^*). \end{aligned}$$

3.2 Quantum Circuits

A quantum circuit takes an n -qubit input $|\Psi\rangle$ together with m *ancilla* qubits initialized to $|0\rangle$ ⁴ and produces an $(n+m)$ -qubit output of which some qubits are designated as outputs and the rest are *garbage*, as in Figure 10.

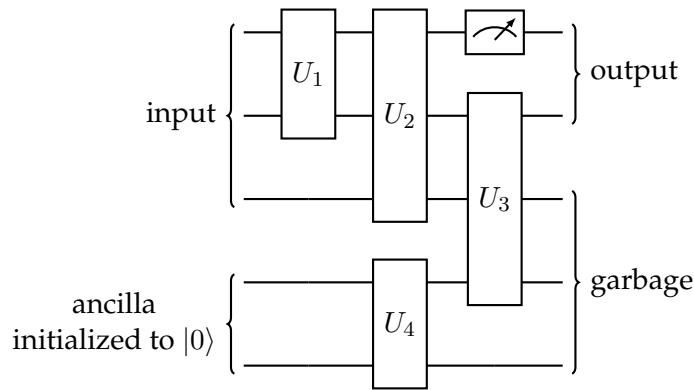


Figure 10: A general quantum circuit with input, ancilla, output, and garbage wires.

⁴Ancilla qubits act as scratch workspace.

3.2.1 2-qubit unitary gates: example

CNOT. The CNOT gate has truth table

Input	Output
$ 00\rangle$	$ 00\rangle$
$ 01\rangle$	$ 01\rangle$
$ 10\rangle$	$ 11\rangle$
$ 11\rangle$	$ 10\rangle$

i.e. $|ab\rangle \xrightarrow{\text{CNOT}} |a(a \oplus b)\rangle$ for $a, b \in \{0, 1\}$. By linearity, for a general $|\Psi\rangle = \sum_{ab} \alpha_{ab} |ab\rangle$,

$$\text{CNOT}(|\Psi\rangle) = \alpha_{00} |00\rangle + \alpha_{01} |01\rangle + \alpha_{10} |11\rangle + \alpha_{11} |10\rangle,$$

so the amplitudes on $|10\rangle$ and $|11\rangle$ are swapped.

3.2.2 3-qubit unitary gates: example

CCNOT (Toffoli). The CCNOT gate maps $|abc\rangle \xrightarrow{\text{CCNOT}} |a, b, c \oplus (a \wedge b)\rangle$.

Exercise 3.1. For $|\Psi\rangle = \sum_{x \in \{0,1\}^3} \alpha_x |x\rangle$, prove that

$$\text{CCNOT}(|\Psi\rangle) = \left(\sum_{x \in \{0,1\}^3 \setminus \{110, 111\}} \alpha_x |x\rangle \right) + \alpha_{111} |110\rangle + \alpha_{110} |111\rangle.$$

Proof. Inspecting the truth table for CCNOT, only the inputs $|110\rangle$ and $|111\rangle$ map to outputs different from themselves; specifically, they swap. By linearity,

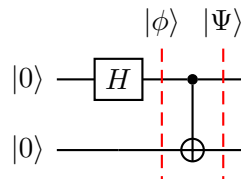
$$\begin{aligned} \text{CCNOT}(|\Psi\rangle) &= \sum_x \alpha_x \text{CCNOT}(|x\rangle) \\ &= \sum_{x \notin \{110, 111\}} \alpha_x |x\rangle + \alpha_{110} |111\rangle + \alpha_{111} |110\rangle, \end{aligned}$$

giving the claimed expression. □

3.2.3 Building Multi-Gate Circuits

We have seen 1-, 2-, and 3-qubit unitaries. Combining them as gates in a quantum circuit allows non-trivial transformations.

Example. Consider the two-qubit circuit



After the Hadamard, the joint state is $|\phi\rangle = H|0\rangle \otimes |0\rangle = \frac{|00\rangle + |10\rangle}{\sqrt{2}}$. After the CNOT,

$$|\Psi\rangle = \text{CNOT}(|\phi\rangle) = \frac{|00\rangle + |11\rangle}{\sqrt{2}}.$$

3.2.4 Entanglement

The output state $|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ from the example above cannot be written as a tensor product $|\phi_0\rangle \otimes |\phi_1\rangle$. Suppose otherwise, with $|\phi_0\rangle = \alpha_0|0\rangle + \beta_0|1\rangle$ and $|\phi_1\rangle = \alpha_1|0\rangle + \beta_1|1\rangle$. Then

$$|\phi_0\rangle \otimes |\phi_1\rangle = \begin{bmatrix} \alpha_0\alpha_1 \\ \alpha_0\beta_1 \\ \beta_0\alpha_1 \\ \beta_0\beta_1 \end{bmatrix} \stackrel{?}{=} \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix},$$

so $\alpha_0\alpha_1 = \frac{1}{\sqrt{2}}$ (hence both nonzero), $\beta_0\beta_1 = \frac{1}{\sqrt{2}}$ (hence both nonzero), and $\alpha_0\beta_1 = 0$, $\beta_0\alpha_1 = 0$ (which forces at least one of β_1, α_0 to be zero, contradicting the previous line).

Definition 4 (Entangled 2-qubit state). A 2-qubit state $|\Psi\rangle$ is *entangled* if it cannot be written as a tensor product of two 1-qubit states.

What does entanglement do? For $|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$, measuring the first qubit in the computational basis yields:

- “0” with probability $\frac{1}{2}$, leaving the post-measurement state $|00\rangle$;
- “1” with probability $\frac{1}{2}$, leaving $|11\rangle$.

In either case the second qubit is determined: measuring the first qubit *instantly* fixes the value of the second.

Remark. The state $|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ is an example of an EPR (Einstein–Podolsky–Rosen) pair, which we revisit in future lectures.

Remark. Consider the alternative $|\Psi'\rangle = |+\rangle \otimes |+\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$. Measuring the first qubit of $|\Psi'\rangle$ in the computational basis leaves a uniform superposition over $|0\rangle, |1\rangle$ in the second qubit, regardless of the outcome — behavior impossible for $|\Psi\rangle$. So $|\Psi'\rangle$ is *not* entangled (it factors), and $|\Psi\rangle$ is.

3.2.5 No-Cloning

The circuit above transforms $|\phi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)$ into $|\Psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$. Inspecting the basis states, one is tempted to see this as “copying the first qubit into the second.” Does there in fact exist a unitary U that clones every state, i.e. such that

$$U(|\Psi\rangle \otimes |0\rangle) = |\Psi\rangle \otimes |\Psi\rangle \quad \text{for every } |\Psi\rangle?$$

The answer is no.

Theorem 3.1 (Weak No-Cloning). *There is no unitary U such that $U(|\Psi\rangle \otimes |0\rangle) = |\Psi\rangle \otimes |\Psi\rangle$ for all single-qubit states $|\Psi\rangle$.*

Proof. Suppose such U existed. Then $U(|0\rangle \otimes |0\rangle) = |00\rangle$ and $U(|1\rangle \otimes |0\rangle) = |11\rangle$. By linearity,

$$\begin{aligned} U\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |0\rangle\right) &= \frac{1}{\sqrt{2}} U(|0\rangle \otimes |0\rangle) + \frac{1}{\sqrt{2}} U(|1\rangle \otimes |0\rangle) \\ &= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle). \end{aligned}$$

But cloning $|\Psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle$ should produce

$$|+\rangle \otimes |+\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle) \neq \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle),$$

a contradiction. □

Key Takeaways. An n -qubit state is a unit vector in \mathbb{C}^{2^n} ; the inner product extends from row-column matrix products, and the outer product gives the corresponding rank-1 matrix. Entangled states cannot be written as tensor products and exhibit non-classical correlations between measurement outcomes. No unitary can copy an arbitrary unknown quantum state — a property we will exploit throughout the cryptographic portion of the course.

3.3 Further Reading

- Inner and outer products with qubits: Lecture 2 of [OW15].
- Quantum circuits: Lectures 1 and 2 of [OW15].
- No-cloning and entanglement: Lecture 3 of [OW15].

4 Quantum Algorithms

4.1 The Toffoli (CCNOT) Gate

The Toffoli (CCNOT) gate is universal for reversible classical computation. When its third input is initialized to $|1\rangle$, it implements a NAND gate reversibly: the output of the third wire is $1 \oplus (a \wedge b) = \text{NAND}(a, b)$.

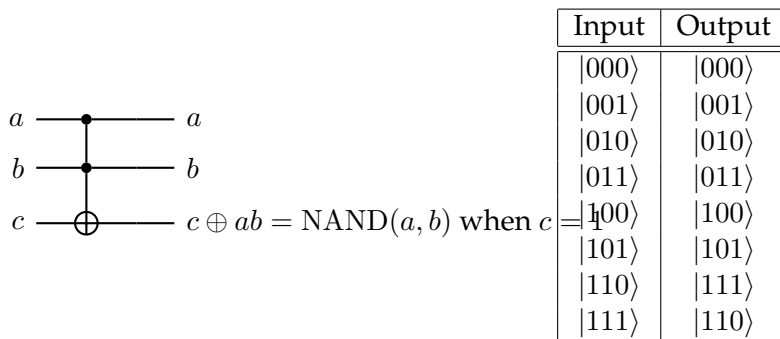


Figure 11: The CCNOT (Toffoli) gate and its truth table.

Remark. For any classical Turing machine computing a function F in time T , there is a classical circuit implementing F with $O(T \log T)$ NAND gates [NC16]. By substituting Toffoli for each NAND (with ancillas), one obtains a reversible circuit with $O(T \log T)$ Toffoli gates.

4.2 Reversible Circuits and Uncomputing

4.2.1 Example: Reversible Circuit for a NAND-NAND Composition

Figure 12 shows a classical NAND-based circuit (top) and a reversible analogue (bottom). The reversible circuit is described by a unitary \mathcal{U} ; running it in reverse corresponds to $\mathcal{U}^{-1} = \mathcal{U}^\dagger$. Note that, in the reversible version, only the final wire carries the output of interest, while the other wires hold “garbage” — intermediate values needed only to preserve reversibility.

Remark. Since classical multiplication has polynomial-size circuits over NAND gates, there is a reversible (Toffoli-based) circuit implementing $\text{MULT}(p, q) \mapsto (N, \text{garbage})$ in $\text{poly}(|p| + |q|)$ gates. The relevance is that quantum implementations of Shor’s algorithm require reversible arithmetic subroutines of this kind; we will use this fact when discussing factoring.

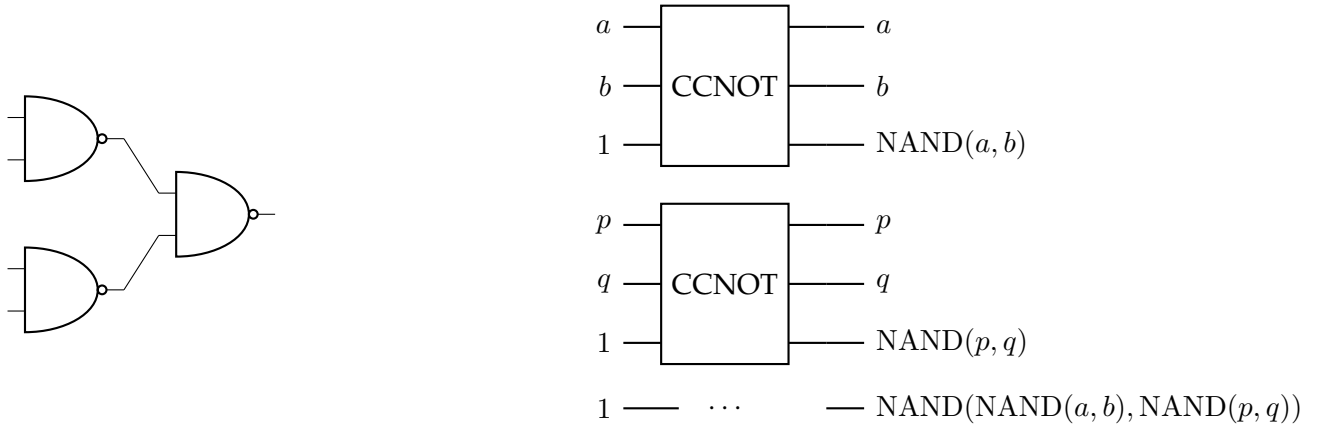


Figure 12: A classical NAND-NAND circuit (left) and its reversible analogue using Toffoli gates (right).

4.2.2 Uncomputing

The circuit in Figure 13 applies a unitary U , copies the desired output onto a fresh ancilla via CNOTs, and then applies U^{-1} to restore the ancillas to their original $|0\rangle$ state. Schematically,

$$|x\rangle \otimes |0\rangle^{\otimes k} \otimes |0\rangle \mapsto |x\rangle \otimes |0\rangle^{\otimes k} \otimes |\text{output}\rangle.$$

After uncomputing, the ancilla register is no longer entangled with the input and output registers. This permits us to “trace out” the ancillas (equivalently, simply discard them), and we may write the overall transformation succinctly as

$$|x\rangle \otimes |0\rangle \mapsto |x\rangle \otimes |\text{output}\rangle.$$

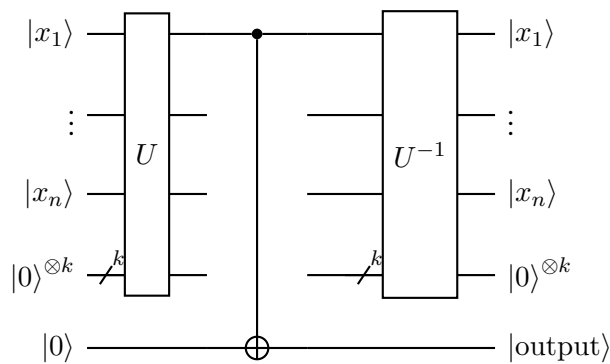


Figure 13: A circuit that computes U , copies its output, then uncomputes the ancillas.

4.3 The Deutsch-Jozsa Algorithm

We now turn to one of the simplest examples illustrating the power of quantum computation: the Deutsch-Jozsa problem.

4.3.1 Quantum Implementation of Classical Functions

A quantum circuit C_f implements a classical function $f: \{0, 1\}^n \rightarrow \{0, 1\}^m$ if, for every $x \in \{0, 1\}^n$ and $y \in \{0, 1\}^m$,

$$C_f(|x\rangle|y\rangle|0\rangle^{\otimes k}) = |x\rangle|y \oplus f(x)\rangle|0\rangle^{\otimes k}.$$

We omit the ancilla register from notation, writing simply $C_f(|x\rangle|y\rangle) = |x\rangle|y \oplus f(x)\rangle$.

4.3.2 The Phase-Kickback Trick

Take $m = 1$, so $f: \{0, 1\}^n \rightarrow \{0, 1\}$. For input $|x\rangle|b\rangle$ we have

$$C_f(|x\rangle|b\rangle) = |x\rangle|b \oplus f(x)\rangle.$$

Now apply C_f to $|x\rangle|-\rangle$, where $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. By linearity,

$$\begin{aligned} C_f(|x\rangle|-\rangle) &= \frac{1}{\sqrt{2}}(C_f(|x\rangle|0\rangle) - C_f(|x\rangle|1\rangle)) \\ &= |x\rangle \otimes \frac{|f(x)\rangle - |\neg f(x)\rangle}{\sqrt{2}}. \end{aligned}$$

Case analysis on $f(x)$:

- If $f(x) = 0$: the output state is $|x\rangle \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} = |x\rangle|-\rangle$.
- If $f(x) = 1$: the output state is $|x\rangle \otimes \frac{|1\rangle - |0\rangle}{\sqrt{2}} = -|x\rangle|-\rangle$.

Combining both cases,

$$C_f(|x\rangle|-\rangle) = (-1)^{f(x)} |x\rangle|-\rangle. \quad (4.1)$$

The value of $f(x)$ has been “kicked back” as a phase on the input register. More generally, by linearity, for any superposition $\sum_x \alpha_x |x\rangle$,

$$C_f\left(\sum_x \alpha_x |x\rangle|-\rangle\right) = \sum_x (-1)^{f(x)} \alpha_x |x\rangle|-\rangle. \quad (4.2)$$

4.3.3 Warm-up: $n = 1$

Suppose $f: \{0, 1\} \rightarrow \{0, 1\}$ is promised to be either *constant* ($f(0) = f(1)$) or *balanced* ($f(0) \neq f(1)$). How many oracle queries are needed to determine which? Classically, 2 queries are necessary and sufficient. Quantum-mechanically, a single query suffices.

Query the oracle on $|+\rangle|-\rangle$:

$$C_f(|+\rangle|-\rangle) = \frac{1}{\sqrt{2}}((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle) \otimes |-\rangle.$$

- If f is constant, the first register is $\pm|+\rangle$.
- If f is balanced, the first register is $\pm|-\rangle$.

Applying a Hadamard gate to the first register and measuring in the computational basis yields 0 if f is constant and 1 if f is balanced.

4.3.4 The General Algorithm

For general n , the algorithm proceeds:

1. Prepare $\sum_{x \in \{0,1\}^n} \frac{|x\rangle}{2^{n/2}} \otimes |-\rangle$, which can be obtained by applying $H^{\otimes n}$ to $|0\rangle^{\otimes n}$ (and using $|-\rangle$ for the ancilla):

$$|0\rangle^{\otimes n} \xrightarrow{H^{\otimes n}} \text{---} \boxed{H^{\otimes n}} \text{---}$$

2. Apply the oracle C_f . By (4.2) the input register becomes

$$|\psi\rangle = \sum_{x \in \{0,1\}^n} \frac{(-1)^{f(x)} |x\rangle}{2^{n/2}}.$$

3. Apply $H^{\otimes n}$ and measure in the computational basis.

We argue:

- If f is constant with value c , then $|\psi\rangle = (-1)^c \frac{1}{2^{n/2}} \sum_x |x\rangle$, and applying $H^{\otimes n}$ yields $(-1)^c |0^n\rangle$. The measurement outcome is 0^n with probability 1.
- If f is balanced, the measurement outcome is *never* 0^n (see exercise below).

Exercise 4.1. Prove that for balanced f ,

$$H^{\otimes n} \left(\sum_x \frac{(-1)^{f(x)} |x\rangle}{2^{n/2}} \right) \neq \pm |0^n\rangle.$$

Hint: the amplitude on $|0^n\rangle$ after the Hadamard is $\frac{1}{2^n} \sum_x (-1)^{f(x)}$. For balanced f , the $+1$ and -1 terms cancel, giving amplitude 0.

Key Takeaways. Reversible classical computation is universal for quantum circuits via the Toffoli gate, at a constant-factor overhead. Uncomputing ancillas keeps them disentangled from the input and output, which is essential for treating quantum subroutines as “black boxes.” The phase kickback trick converts function values into amplitudes, enabling interference patterns to distinguish global properties — as in Deutsch-Jozsa, which solves a problem requiring 2 classical queries with just 1 quantum query.

4.4 Further Reading

- Grover’s algorithm: Lecture 4 of [OW15].
- Quantum query complexity: Lecture 5 of [OW15].
- The Deutsch-Jozsa algorithm: S1.4.4 of [NC16].

5 Quantum Fourier Transform

We continue our study of quantum algorithms, beginning where we left off with the Deutsch-Jozsa setup: superposition access to a Boolean function $f: \{0, 1\}^n \rightarrow \{0, 1\}$ via an oracle C_f . Recall the phase-kickback identity $C_f(|x\rangle|-\rangle) = (-1)^{f(x)}|x\rangle|-\rangle$, which extends by linearity to

$$\sum_x \alpha_x |x\rangle|-\rangle \xrightarrow{C_f} \sum_x (-1)^{f(x)} \alpha_x |x\rangle|-\rangle.$$

The Deutsch-Jozsa promise: f is either constant or balanced, and we wish to decide which using a single quantum query.

5.1 The Deutsch-Jozsa Algorithm

The algorithm proceeds as follows:

1. Prepare the uniform superposition $\frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} |x\rangle$ via $H^{\otimes n} |0^n\rangle$.
2. Apply the oracle on $(\frac{1}{2^{n/2}} \sum_x |x\rangle)|-\rangle$ to obtain

$$\frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} (-1)^{f(x)} |x\rangle|-\rangle.$$

3. Discard the second register, apply $H^{\otimes n}$ to the first register, and measure it in the computational basis.

Theorem 5.1. f is constant if and only if the measurement outcome equals 0^n .

Proof. We prove both directions.

(\Rightarrow) **f constant implies outcome 0^n .** If f is constant, $(-1)^{f(x)} = \pm 1$ for all x , so

$$\sum_x (-1)^{f(x)} H^{\otimes n} |x\rangle = \pm \sum_x H^{\otimes n} |x\rangle = \pm 2^{n/2} |0^n\rangle.$$

After normalization, this is $\pm |0^n\rangle$, and measurement always outputs 0^n .

(\Leftarrow) **Outcome 0^n implies f constant.** We show the contrapositive: if f is balanced, the amplitude on $|0^n\rangle$ after the Hadamard is zero, so 0^n cannot be observed.

Recall that for $x \in \{0, 1\}^n$,

$$H^{\otimes n} |x\rangle = \bigotimes_{i \in [n]} \frac{1}{\sqrt{2}} (|0\rangle + (-1)^{x_i} |1\rangle) = \frac{1}{2^{n/2}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle,$$

where $x \cdot y = \bigoplus_i x_i y_i$ is the inner product over \mathbb{F}_2 . Substituting,

$$\sum_x (-1)^{f(x)} H^{\otimes n} |x\rangle = \frac{1}{2^{n/2}} \sum_{x,y} (-1)^{f(x)+x \cdot y} |y\rangle.$$

The amplitude on $|0^n\rangle$ is thus (up to normalization)

$$\sum_x (-1)^{f(x)+0} = \sum_x (-1)^{f(x)}.$$

For balanced f , exactly half of the 2^n terms are $+1$ and half are -1 , so the sum is 0. \square

In summary, a single quantum query, sandwiched between Hadamard transforms, distinguishes constant from balanced. Outcome 0^n means constant; any other outcome means balanced.

5.2 Fourier Transform over \mathbb{F}_2^n

For $x, \sigma \in \mathbb{F}_2^n$, define their inner product

$$\sigma \cdot x = \bigoplus_{i:\sigma_i=1} x_i \in \mathbb{F}_2,$$

i.e. the parity of x restricted to coordinates where σ is 1.

Definition 5 (Fourier basis on \mathbb{F}_2^n). For each $\sigma \in \{0, 1\}^n$, define the function

$$\chi'_\sigma(x) = (-1)^{\sigma \cdot x} = \prod_{i:\sigma_i=1} (-1)^{x_i}$$

and the unit-length column vector (in \mathbb{C}^{2^n})

$$\chi_\sigma = \frac{1}{2^{n/2}} [\chi'_\sigma(0^n), \chi'_\sigma(0^{n-1}1), \dots, \chi'_\sigma(1^n)]^\top.$$

The set $\{\chi_\sigma\}_{\sigma \in \{0,1\}^n}$ is the *Fourier basis*.

Theorem 5.2. $\{\chi_\sigma\}_{\sigma \in \{0,1\}^n}$ is an orthonormal basis of \mathbb{C}^{2^n} .

Proof. Since there are 2^n such vectors and they have unit norm by construction, it suffices to show they are pairwise orthogonal: $\langle \chi_\sigma | \chi_\gamma \rangle = 0$ whenever $\sigma \neq \gamma$, and $= 1$ when $\sigma = \gamma$.

Computing,

$$\begin{aligned} \langle \chi_\sigma | \chi_\gamma \rangle &= \frac{1}{2^n} \sum_x \chi'_\sigma(x) \chi'_\gamma(x) \\ &= \frac{1}{2^n} \sum_x (-1)^{\sigma \cdot x} (-1)^{\gamma \cdot x} \\ &= \frac{1}{2^n} \sum_x (-1)^{(\sigma \oplus \gamma) \cdot x} \\ &= \mathbb{E}_x [\chi'_{\sigma \oplus \gamma}(x)]. \end{aligned}$$

If $\sigma \oplus \gamma = 0^n$, every summand is 1, so the expectation is 1. Otherwise, $\sigma \oplus \gamma \neq 0^n$, and the values $(\sigma \oplus \gamma) \cdot x$ are 0 for exactly half of the x and 1 for the other half (since the map $x \mapsto (\sigma \oplus \gamma) \cdot x$ is a non-zero linear functional on \mathbb{F}_2^n). The $+1$'s and -1 's cancel, giving expectation 0. \square

5.3 Simon's Algorithm

Definition 6 (Simon's promise). A function $f: \{0, 1\}^n \rightarrow \{0, 1\}^n$ is *Simon-promising* if it is two-to-one and there exists a fixed $s \neq 0^n$ such that

$$f(x) = f(y) \iff y = x \oplus s.$$

The goal: given oracle access to C_f for such an f , find s .

Algorithm.

1. Prepare $\frac{1}{2^{n/2}} \sum_x |x\rangle |0^n\rangle$ via $H^{\otimes n}$ on the first n qubits.
2. Apply U_f to obtain $\frac{1}{2^{n/2}} \sum_x |x\rangle |f(x)\rangle$.
3. Measure the second register. Conditional on the outcome $|y\rangle$ for some y in the image of f , the first register collapses to $\frac{1}{\sqrt{2}}(|x'\rangle + |x' \oplus s\rangle)$ for the two preimages of y .
4. Apply $H^{\otimes n}$ to the first register and measure in the computational basis.

Analysis of step 4.

$$\begin{aligned} H^{\otimes n}(|x'\rangle + |x' \oplus s\rangle) &= \sum_{\gamma} (-1)^{x' \cdot \gamma} |\gamma\rangle + \sum_{\gamma} (-1)^{(x' \oplus s) \cdot \gamma} |\gamma\rangle \\ &= \sum_{\gamma} (-1)^{x' \cdot \gamma} (1 + (-1)^{s \cdot \gamma}) |\gamma\rangle. \end{aligned}$$

The factor $1 + (-1)^{s \cdot \gamma}$ vanishes when $s \cdot \gamma = 1$ and equals 2 when $s \cdot \gamma = 0$. Therefore measurement returns a uniformly random γ in the hyperplane $s^\perp = \{\gamma : s \cdot \gamma = 0\}$.

Recovering s . Each run yields a uniformly random $\gamma \in s^\perp$. After $\Omega(n)$ runs we will, with constant probability, have $n - 1$ linearly independent vectors in s^\perp , which determine s uniquely up to its single missing coordinate (recoverable by trying both options).

Success probability. The hyperplane s^\perp has size 2^{n-1} . Given the previous k vectors $\gamma_1, \dots, \gamma_k$, the probability that γ_{k+1} lies in their span is at most $\frac{2^k}{2^{n-1}}$. Hence the probability that all $n - 1$ are linearly independent is

$$\prod_{k=0}^{n-2} \left(1 - \frac{2^k}{2^{n-1}}\right) = \prod_{i=1}^{n-1} \left(1 - \frac{1}{2^i}\right).$$

This product is lower-bounded by a constant, e.g. $\geq \frac{1}{4}$ [OW15]. So Simon's algorithm succeeds with probability $\geq \frac{1}{4}$ in n iterations, and the success probability can be amplified to $1 - \text{negl}(n)$ by $O(n)$ further iterations.

Key Takeaways. The Deutsch-Jozsa algorithm uses interference — realized as a Hadamard sandwich around an oracle call — to distinguish constant from balanced functions in a single query. The same Hadamard-oracle-Hadamard pattern, with a different post-processing step, solves Simon's problem in $O(n)$ queries (versus the $\Omega(2^{n/2})$ classical lower bound). The Fourier basis on \mathbb{F}_2^n formalizes how Hadamards convert structural information about f into amplitudes.

5.4 Further Reading

- Deutsch-Jozsa: S1.4.4 of [NC16]; Lecture 5 S4.4 of [OW15].
- Quantum Fourier transform: Lectures 6, 7 of [OW15].
- Simon's problem and algorithm: Lecture 6 S2.4 of [OW15].

6 Fourier Basis, Simon's Algorithm, and Period Finding

We have so far seen how quantum circuits can implement classical functions, and explored gate sets that act on quantum states. In this lecture, we adopt a complementary viewpoint: *functions themselves* can be naturally represented as quantum states. This perspective sets up a clean expression of Simon's algorithm and motivates the move from Boolean Fourier analysis to period finding on \mathbb{Z}_N .

6.1 Functions as Quantum States

Let $g: \{0, 1\}^n \rightarrow \mathbb{C}$. We may represent g as a complex column vector of length 2^n :

$$g = \begin{bmatrix} g(0^n) \\ g(0^{n-1}1) \\ \vdots \\ g(1^n) \end{bmatrix}.$$

If g is normalized so that $\sum_x |g(x)|^2 = 1$, then this vector is a valid quantum state, with $g(x)$ playing the role of the amplitude on $|x\rangle$:

$$|g\rangle = \sum_{x \in \{0,1\}^n} g(x) |x\rangle.$$

For convenience, we frequently suppress normalization. The unnormalized state $|g\rangle = \sum_x g(x) |x\rangle$ then represents the function up to a global scaling.

6.1.1 Functions in the Fourier Basis

Recall the Fourier basis $\{|\chi_s\rangle\}_{s \in \{0,1\}^n}$ on \mathbb{F}_2^n , where, in normalized vector form,

$$\chi_s = \frac{1}{\sqrt{2^n}} \begin{bmatrix} \chi_s(0^n) \\ \chi_s(0^{n-1}1) \\ \vdots \\ \chi_s(1^n) \end{bmatrix}, \quad \chi_s(x) = (-1)^{s \cdot x}.$$

Since $\{|\chi_s\rangle\}_s$ is an orthonormal basis of \mathbb{C}^{2^n} , every state $|g\rangle$ can be expanded as

$$|g\rangle = \sum_s \hat{g}(s) |\chi_s\rangle,$$

where the Fourier coefficient $\hat{g}(s)$ is the inner product of $|g\rangle$ with $|\chi_s\rangle$:

$$\hat{g}(s) = \langle \chi_s | g \rangle = \sum_x \chi_s(x) g(x) = \mathbb{E}_{x \in \{0,1\}^n} [\chi_s(x) g(x)],$$

where the last equality uses the normalizing factor of $1/2^n$ implicit in χ_s and turns the sum into an expectation. In the special case $s = 0^n$, we have $\chi_0(x) = 1$ for all x , so

$$\widehat{g}(0) = \mathbb{E}_x[g(x)].$$

6.2 Revisiting Deutsch-Jozsa via Fourier Analysis

Recall the Deutsch-Jozsa setup: an oracle C_f for a function $f: \{0, 1\}^n \rightarrow \{0, 1\}$ that is promised to be either constant or balanced. The output of querying C_f on the uniform superposition $\frac{1}{\sqrt{2^n}} \sum_x |x\rangle |-\rangle$, and then discarding $|-\rangle$, is

$$\frac{1}{\sqrt{2^n}} \sum_x (-1)^{f(x)} |x\rangle.$$

Define $g(x) = (-1)^{f(x)}$. Then this is precisely the state $|g\rangle$, which we can expand in the Fourier basis as

$$|g\rangle = \sum_s \widehat{g}(s) |\chi_s\rangle, \quad \widehat{g}(s) = \mathbb{E}_x[\chi_s(x)g(x)] = \mathbb{E}_x[\chi_s(x)(-1)^{f(x)}].$$

Case 1: f is constant. Say $f(x) = b$ for all x . Then $g(x) = (-1)^b$ is itself constant, so

$$\widehat{g}(s) = (-1)^b \cdot \mathbb{E}_x[\chi_s(x)] = \begin{cases} (-1)^b & \text{if } s = 0^n, \\ 0 & \text{otherwise.} \end{cases}$$

Thus $|g\rangle = \pm |\chi_{0^n}\rangle$.

Case 2: f is balanced. Exactly half of the values $f(x)$ are 0 and half are 1, so $g(x) = (-1)^{f(x)}$ takes the values $+1$ and -1 on equal-size sets. In particular,

$$\widehat{g}(0^n) = \mathbb{E}_x[g(x)] = 0,$$

i.e. the $|\chi_{0^n}\rangle$ component vanishes, and $|g\rangle$ is a linear combination of the other $|\chi_s\rangle$.

The Hadamard step. The Deutsch-Jozsa algorithm finishes by applying $H^{\otimes n}$ and measuring. We claim this reads out the Fourier coefficients directly.

Theorem 6.1. $H^{\otimes n}(\sum_s \widehat{g}(s) |\chi_s\rangle) = \sum_s \widehat{g}(s) |s\rangle$.

Proof. Recall $|\chi_s\rangle = \sum_y (-1)^{s \cdot y} |y\rangle$ (up to normalization), and $H^{\otimes n} |y\rangle = \sum_z (-1)^{y \cdot z} |z\rangle$ (up to normalization). Substituting,

$$\begin{aligned} H^{\otimes n} \left(\sum_s \widehat{g}(s) |\chi_s\rangle \right) &= \sum_s \widehat{g}(s) \sum_y (-1)^{s \cdot y} H^{\otimes n} |y\rangle \\ &= \sum_s \widehat{g}(s) \sum_{y,z} (-1)^{s \cdot y + y \cdot z} |z\rangle \\ &= \sum_s \widehat{g}(s) \sum_z |z\rangle \sum_y (-1)^{y \cdot (s \oplus z)}. \end{aligned}$$

The inner sum $\sum_y (-1)^{y \cdot (s \oplus z)}$ equals 2^n if $s = z$ and 0 otherwise. Hence

$$H^{\otimes n} \left(\sum_s \widehat{g}(s) |\chi_s\rangle \right) = \sum_s \widehat{g}(s) |s\rangle. \quad \square$$

In Case 1, the post-Hadamard state is $\pm |0^n\rangle$, and measurement always outputs 0^n . In Case 2, the amplitude on $|0^n\rangle$ is zero, so 0^n is never observed. The Fourier-basis viewpoint makes this discrimination immediate.

6.3 Simon's Algorithm Revisited

Simon's promise: $f: \{0,1\}^n \rightarrow \{0,1\}^m$ is two-to-one with $f(x) = f(x \oplus c)$ for some unknown $c \neq 0^n$.

Setup. Prepare $\sum_x |x\rangle |0^m\rangle$, apply U_f to obtain $\sum_x |x\rangle |f(x)\rangle$, and measure the second register. The first register collapses to the unnormalized state

$$|g\rangle = |x'\rangle + |x' \oplus c\rangle,$$

where $x', x' \oplus c$ are the two preimages of the measured outcome.

Fourier-basis analysis. The Fourier coefficient at s is

$$\begin{aligned} \widehat{g}(s) &= \chi_s(x') + \chi_s(x' \oplus c) \\ &= (-1)^{s \cdot x'} + (-1)^{s \cdot (x' \oplus c)} \\ &= (-1)^{s \cdot x'} (1 + (-1)^{s \cdot c}). \end{aligned}$$

The factor $1 + (-1)^{s \cdot c}$ vanishes if $s \cdot c = 1$ and equals 2 if $s \cdot c = 0$. Hence $|g\rangle$ is supported in the Fourier basis exactly on $s \in c^\perp = \{s : s \cdot c = 0\}$.

Read-out. Applying $H^{\otimes n}$ converts $\sum_{s \in c^\perp} \widehat{g}(s) |\chi_s\rangle$ to $\sum_{s \in c^\perp} \widehat{g}(s) |s\rangle$ by Theorem 6.1, and measuring yields a uniformly random $s \in c^\perp$. Collecting $n - 1$ such linearly independent vectors and solving the resulting linear system determines c .

6.4 Fourier Transform over \mathbb{Z}_N

We now move from \mathbb{F}_2^n to \mathbb{Z}_N , where $N = 2^n$. The Fourier basis on \mathbb{Z}_N is

$$\{\chi_0, \chi_1, \dots, \chi_{N-1}\}, \quad \chi_s(x) = \omega^{sx}, \quad s, x \in [0, N-1],$$

where $\omega = \omega_N = e^{2\pi i/N}$ is a primitive N -th root of unity.

6.4.1 Geometry of ω_N

The complex numbers $1, \omega, \omega^2, \dots, \omega^{N-1}$ divide the unit circle into N equal arcs, each of angle $2\pi/N$:

$$\omega_N = e^{2\pi i/N} = \cos\left(\frac{2\pi}{N}\right) + i \sin\left(\frac{2\pi}{N}\right).$$

In particular, $\omega^N = e^{2\pi i} = 1$.

6.4.2 Properties of χ_s

- χ_s corresponds to the column vector $[\chi_s(0), \chi_s(1), \dots, \chi_s(N-1)]^\top$.
- $\chi_0(x) = \omega^0 = 1$ for all x .
- Symmetry: $\chi_s(x) = \chi_x(s) = \omega^{sx}$.
- Average over x :

$$\mathbb{E}_{x \in \mathbb{Z}_N}[\chi_s(x)] = \frac{1}{N} \sum_{x=0}^{N-1} \omega^{sx} = \begin{cases} 1 & s = 0, \\ 0 & s \neq 0, \end{cases}$$

where the second case uses the geometric series identity $\sum_{x=0}^{N-1} \omega^{sx} = \frac{\omega^{sN} - 1}{\omega^s - 1} = 0$ for $s \neq 0$ (since $\omega^{sN} = 1$).

- Multiplicativity: $\chi_s(x)\chi_r(x) = \chi_{s+r}(x)$.
- Conjugate: $(\chi_r(x))^* = \omega^{-rx} = \chi_{-r}(x) = \chi_r(-x)$. Unlike the \mathbb{F}_2^n setting, $\chi_r(x) \in \mathbb{C}$ in general, not $\{\pm 1\}$.
- Orthonormality:

$$\langle \chi_s | \chi_r \rangle = \mathbb{E}_x[(\chi_s(x))^* \chi_r(x)] = \mathbb{E}_x[\chi_{r-s}(x)] = \begin{cases} 1 & r = s, \\ 0 & r \neq s. \end{cases}$$

- Hence $\{\chi_s\}$ is an orthonormal basis of \mathbb{C}^N .
- Fourier expansion: any state $|g\rangle = \sum_s \hat{g}(s) |\chi_s\rangle$, where

$$\hat{g}(s) = \langle \chi_s | g \rangle = \mathbb{E}_x[\chi_{-s}(x)g(x)].$$

Note the conjugate χ_{-s} in place of χ_s ; over \mathbb{Z}_N these are no longer the same.

6.5 Period Finding (Preview)

Let $f: \mathbb{Z}_N \rightarrow S$ be *periodic* with period s :

$$\forall x, f(x) = f(x+s) = f(x+2s) = \dots,$$

equivalently, $f(x) = f(y)$ iff $|y-x|$ is a multiple of s . This implies $s \mid N$. Our goal: find s .

The procedure parallels Simon's algorithm, with two key differences: f is no longer two-to-one (each value has N/s preimages), and the Hadamard transform is replaced by the QFT over \mathbb{Z}_N . We work through the algorithm in detail in the next lecture.

Key Takeaways. Treating a function $g: \{0, 1\}^n \rightarrow \mathbb{C}$ as a quantum state $|g\rangle$ and changing to the Fourier basis gives a uniform way to analyze Deutsch-Jozsa and Simon's algorithm: in both cases, the support of \hat{g} encodes the structural property we want to read out. Shifting from \mathbb{F}_2^n to \mathbb{Z}_N requires only minor changes (complex-valued characters, geometric-series cancellation in place of ± 1 parity), and sets up the period-finding problem at the heart of Shor's algorithm.

6.6 Further Reading

- Boolean Fourier analysis and Simon's algorithm: Lectures 6, 7 of [OW15].
- Simon's problem and algorithm: Lecture 6 S2.4 of [OW15].

7 Period-finding and Shor's Algorithm

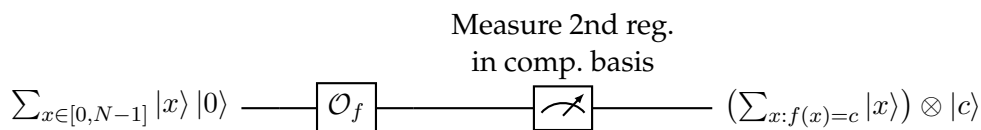
In the previous lecture, we began studying periodic functions and how quantum resources can be used to efficiently extract a hidden period. Concretely, let $f: \mathbb{Z}_N \rightarrow \mathcal{Y}$ be a function for which there exists s such that, for all x , $f(x) = f(x + s) = f(x + 2s) = \dots$. Equivalently,

$$f(x) = f(y) \iff |x - y| \text{ is a multiple of } s.$$

In particular this implies $s \mid N$. Simon's algorithm finds the hidden s with $O(\log N)$ quantum queries (i.e. polynomial in the bit-length of N) under the assumption that f is two-to-one. In this lecture we extend that approach to general periodic functions on \mathbb{Z}_N and then apply it to factor integers via Shor's algorithm.

7.1 Period-Finding

As in Simon's algorithm, we prepare a uniform superposition over the domain, apply the oracle \mathcal{O}_f , and measure the second register in the computational basis (ignoring normalization for clarity):



Let $|g\rangle = \sum_{x: f(x)=c} |x\rangle$ denote the (unnormalized) state of the first register after measurement. Because f has period s , the preimages of c form an arithmetic progression, so

$$|g\rangle = |x'\rangle + |x' + s\rangle + |x' + 2s\rangle + \dots + |x' + (N/s - 1)s\rangle$$

for some x' . We may equivalently regard g as the function on \mathbb{Z}_N whose value $g(x)$ is the amplitude on $|x\rangle$:

$$|g\rangle = \sum_x g(x) |x\rangle.$$

Switching to the Fourier basis. Recall that the Fourier basis on \mathbb{Z}_N consists of $\{|\chi_\sigma\rangle\}_{\sigma \in [0, N-1]}$ with $\chi_\sigma(x) = \omega^{\sigma x}$, where $\omega = e^{2\pi i/N}$. Writing $|g\rangle = \sum_\sigma \widehat{g}(\sigma) |\chi_\sigma\rangle$, the Fourier coefficients are

$$\begin{aligned}\widehat{g}(\sigma) &= \mathbb{E}_x [\chi_\sigma(x)^* g(x)] \\ &= \mathbb{E}_x [\omega^{-\sigma x} g(x)] \\ &= \frac{s}{N} \sum_{k=0}^{N/s-1} \omega^{-\sigma(x'+ks)} \\ &= \frac{s}{N} \omega^{-\sigma x'} \sum_{k=0}^{N/s-1} \omega^{-\sigma ks},\end{aligned}$$

since $g(x)$ is nonzero only on the N/s points of the form $x' + ks$.⁵

We now analyze $\widehat{g}(\sigma)$ in two cases.

- **Case 1:** $\sigma s \equiv 0 \pmod{N}$. Every term $\omega^{-\sigma ks}$ equals 1, so the geometric sum has N/s terms:

$$\widehat{g}(\sigma) = \frac{s}{N} \omega^{-\sigma x'} \cdot \frac{N}{s} = \omega^{-\sigma x'} \quad (\text{up to normalization}).$$

- **Case 2:** $\sigma s \not\equiv 0 \pmod{N}$. The sum is a non-trivial geometric series:

$$\begin{aligned}\sum_{k=0}^{N/s-1} \omega^{-\sigma ks} &= \frac{(\omega^{-\sigma s})^{N/s} - 1}{\omega^{-\sigma s} - 1} \\ &= \frac{\omega^{-\sigma N} - 1}{\omega^{-\sigma s} - 1} \\ &= \frac{1 - 1}{\omega^{-\sigma s} - 1} = 0,\end{aligned}$$

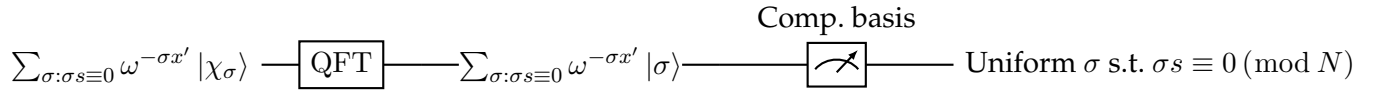
where the second equality uses $\omega^N = 1$. Therefore $\widehat{g}(\sigma) = 0$.

Theorem 7.1. *The Fourier transform $\widehat{g}(\sigma)$ is supported exactly on those σ for which $\sigma s \equiv 0 \pmod{N}$.*

By Theorem 7.1, after the measurement of the second register we can write

$$|g\rangle = \sum_{\sigma: \sigma s \equiv 0 \pmod{N}} \omega^{-\sigma x'} |\chi_\sigma\rangle.$$

Applying the quantum Fourier transform takes $|\chi_\sigma\rangle \rightarrow |\sigma\rangle$, and now measurement in the computational basis yields a uniformly random σ such that $\sigma s \equiv 0 \pmod{N}$:



⁵The original notes wrote this expectation as $\mathbb{E}_k[\cdot]$. We have rewritten the manipulation explicitly: $\mathbb{E}_x[\cdot]$ averages over N values of x , of which exactly N/s contribute, giving the factor s/N .

Recovering the period. The constraint $\sigma s \equiv 0 \pmod{N}$ is equivalent to $\sigma \in \{0, N/s, 2N/s, \dots, (s-1)N/s\}$. Setting $p = N/s$, the measurement outputs a uniform multiple of p in this set.

Now we use the elementary number-theoretic fact

$$\gcd(a, b) = 1 \implies \gcd(ap, bp) = p.$$

Repeating the experiment twice yields independent samples $\sigma_1 = ap$ and $\sigma_2 = bp$; if $\gcd(a, b) = 1$ then $\gcd(\sigma_1, \sigma_2) = p$, which can be computed efficiently with the Euclidean algorithm. It can be shown that $O(\log N)$ repetitions suffice to obtain two coprime multipliers with high probability; we omit the proof.

7.2 Shor's Algorithm

We now apply period-finding to factor integers. Let $M = pq$ be a product of two n -bit primes; the goal is to recover p and q .

Theorem 7.2. *Suppose we can find $r \in \mathbb{Z}_M$ with $r \not\equiv \pm 1 \pmod{M}$ and $r^2 \equiv 1 \pmod{M}$. Then $\gcd(r-1, M)$ and $\gcd(r+1, M)$ together yield a non-trivial factorization of M .*

Proof. The condition $r^2 \equiv 1 \pmod{M}$ rearranges to $(r-1)(r+1) \equiv 0 \pmod{M}$, i.e. $(r-1)(r+1) = kM$ for some integer $k \neq 0$. Since $r \not\equiv \pm 1$, neither $r-1$ nor $r+1$ is congruent to 0 \pmod{M} . We claim that at least one of $\gcd(r-1, M)$ and $\gcd(r+1, M)$ is a non-trivial divisor of M .

Suppose for contradiction that both equal 1. Since $(r-1) \mid kM$ and $\gcd(r-1, M) = 1$, we have $(r-1) \mid k$; similarly $(r+1) \mid k$. But these two cannot both divide k in general (since $\gcd(r-1, r+1) \in \{1, 2\}$, the product $(r-1)(r+1)$ divides $2k$, contradicting $(r-1)(r+1) = kM$ for $M \geq 3$ unless $r \equiv \pm 1$). Hence at least one of $\gcd(r \pm 1, M)$ is a non-trivial factor.⁶ \square

Remark. Reducing factoring to finding such an r , plus the period-finding subroutine above, is the structure of Shor's algorithm.

The remaining task is to produce such an r . Sample $A \in \mathbb{Z}_M^*$ uniformly at random and let s be the smallest positive integer with $A^s \equiv 1 \pmod{M}$ (i.e. $s = \text{ord}_M(A)$). If s is even, set $r = A^{s/2}$; then $r^2 = A^s \equiv 1 \pmod{M}$. It is a classical result that for a random $A \in \mathbb{Z}_M^*$, the order s is even and $A^{s/2} \not\equiv \pm 1 \pmod{M}$ with probability at least $\frac{1}{2}$. So after $O(1)$ retries we obtain a usable r .

Thus the problem reduces to:

Given A, M each n -bit, find s with $A^s \equiv 1 \pmod{M}$.

Define $f(x) = A^x \pmod{M}$. Then f is periodic with period s :

$$f(x+s) = A^{x+s} \pmod{M} = (A^x \cdot A^s) \pmod{M} = A^x \pmod{M} = f(x).$$

Hence we can apply the period-finding procedure from Section 7.1 to f .

⁶The cleanest version of this step (and the only step that uses p, q prime) goes: since $M = pq$, the equation $(r-1)(r+1) \equiv 0 \pmod{pq}$ forces $p \mid (r-1)(r+1)$, hence $p \mid r-1$ or $p \mid r+1$; likewise for q . The hypothesis $r \not\equiv \pm 1 \pmod{M}$ rules out both factors hitting the same side, so one side picks up p and the other picks up q .

Summary of the algorithm.

1. Sample $A \in \mathbb{Z}_M^*$ uniformly. If $\gcd(A, M) > 1$, output the factor and stop.
2. Use period-finding to obtain the order $s = \text{ord}_M(A)$.
3. If s is even and $r := A^{s/2} \not\equiv \pm 1 \pmod{M}$, output $\gcd(r-1, M)$ and $\gcd(r+1, M)$; otherwise restart.

From measurement outcome to s . The period-finding step outputs $\gamma = kN/s$ for some random k . The fraction $\gamma/N = k/s$ may then be recovered using the *continued fractions* algorithm, which finds the unique reduced fraction with denominator $\leq \sqrt{N}$ matching γ/N . The denominator is s . We omit the standard analysis.

7.3 The Hidden Subgroup Problem

Both Simon's algorithm and period-finding are instances of a more general framework, the *Hidden Subgroup Problem* (HSP).

Definition 7 (Hidden Subgroup Problem). Let G be a finite group and $H \leq G$ a subgroup. A function $f: G \rightarrow S$ *hides* H if, for all $x, y \in G$,

$$f(x) = f(y) \iff x^{-1}y \in H.$$

Given oracle access to f , the goal is to find a generating set for H .

Simon's algorithm is HSP over $G = \mathbb{Z}_2^n$ with $H = \{0, s\}$; period-finding is HSP over $G = \mathbb{Z}_N$ with $H = s\mathbb{Z}_N$. The quantum Fourier transform over G is the central tool in both cases.

Application: discrete logarithm. Let p be a prime, g a generator of \mathbb{Z}_p^* , and $h = g^s \pmod{p}$ for some unknown $s \in \mathbb{Z}_{p-1}$. Define

$$f: \mathbb{Z}_{p-1}^2 \rightarrow \mathbb{Z}_p^*, \quad f(a, b) = g^a h^{-b} \pmod{p}.$$

We claim that f hides the subgroup $H = \langle (s, 1) \rangle$, since

$$f(a + s, b + 1) = g^{a+s} h^{-b-1} = g^a h^{-b} \cdot (g^s h^{-1}) = g^a h^{-b} \cdot 1 = f(a, b).$$

Solving HSP for this f recovers a generator of H , namely $(s, 1)$, yielding the discrete logarithm.

7.4 Implementing QFT over \mathbb{Z}_N

The quantum Fourier transform on \mathbb{Z}_N acts as

$$|x\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{\sigma=0}^{N-1} \omega^{-\sigma x} |\sigma\rangle,$$

where $\omega = e^{2\pi i/N}$. Concretely, for $N = 16$ and $x = x_3x_2x_1x_0$ (a 4-bit input),

$$|x\rangle \mapsto \frac{1}{4} (|0000\rangle + \omega^{-x} |0001\rangle + \omega^{-2x} |0010\rangle + \dots + \omega^{-15x} |1111\rangle).$$

We will see in the next lecture how to implement this transformation efficiently as a quantum circuit.

Key Takeaways. For a periodic function f on \mathbb{Z}_N with period s , measuring after applying the oracle and a QFT produces a uniform multiple of N/s ; two such samples generally determine s via gcd. Combining this with the reduction “factoring \rightarrow order-finding” (Theorem 7.2) yields Shor’s polynomial-time quantum factoring algorithm. Both Simon’s algorithm and period-finding are instances of the Hidden Subgroup Problem, which also captures discrete logarithm.

7.5 Further Reading

- Boolean Fourier analysis and Simon’s algorithm: Lectures 6, 7 from [OW15].
- Simon’s problem and algorithm: Lecture 6 S2.4 of [OW15].
- Quantum Fourier transform over \mathbb{Z}_N : Lecture 7 of [OW15].

8 Implementation of the QFT and Introduction to Mixed States

In the previous lecture, we saw that the quantum Fourier transform over \mathbb{Z}_2^n is simply $H^{\otimes n}$, and we introduced the QFT over \mathbb{Z}_N for N a power of 2. We now show how to implement the QFT over \mathbb{Z}_N as an efficient quantum circuit, illustrating with $N = 16$.

8.1 QFT for $N = 16$

We seek a circuit that performs

$$|x\rangle = |x_3x_2x_1x_0\rangle \mapsto \frac{1}{4} \sum_{\sigma=0}^{15} \omega^{-\sigma x} |\sigma\rangle, \quad (8.1)$$

where $\omega = e^{2\pi i/16}$ is a primitive 16-th root of unity and $x_3x_2x_1x_0$ is the binary representation of x (with x_3 the most significant bit).

Tensor product factorization. The crucial observation is that the right-hand side of (8.1) factors as an unentangled product of four single-qubit states. Writing $\sigma = 8\sigma_3 + 4\sigma_2 + 2\sigma_1 + \sigma_0$ in binary, we have

$$\omega^{-\sigma x} = \omega^{-(8\sigma_3+4\sigma_2+2\sigma_1+\sigma_0)x} = (\omega^{-8x})^{\sigma_3} (\omega^{-4x})^{\sigma_2} (\omega^{-2x})^{\sigma_1} (\omega^{-x})^{\sigma_0}.$$

Substituting into (8.1) and writing $|\sigma\rangle = |\sigma_3\rangle |\sigma_2\rangle |\sigma_1\rangle |\sigma_0\rangle$,

$$\begin{aligned} \frac{1}{4} \sum_{\sigma} \omega^{-\sigma x} |\sigma\rangle &= \bigotimes_{k=0}^3 \frac{1}{\sqrt{2}} \sum_{\sigma_k \in \{0,1\}} (\omega^{-2^k x})^{\sigma_k} |\sigma_k\rangle \\ &= |\gamma_3\rangle \otimes |\gamma_2\rangle \otimes |\gamma_1\rangle \otimes |\gamma_0\rangle, \end{aligned}$$

where

$$|\gamma_j\rangle = \frac{|0\rangle + \omega^{-2^{3-j}x} |1\rangle}{\sqrt{2}}.$$

Explicitly,

$$\begin{aligned} |\gamma_3\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + \omega^{-8x} |1\rangle), \\ |\gamma_2\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + \omega^{-4x} |1\rangle), \\ |\gamma_1\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + \omega^{-2x} |1\rangle), \\ |\gamma_0\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + \omega^{-x} |1\rangle). \end{aligned}$$

Remark. $\omega^{16} = 1$ and $\omega^8 = -1$. The original notes contain an equation block of the form “ $z = \omega^{-1}$, $y = \omega^{-2}$, $yz = \omega^{-3}$, \dots , $wxyz = \omega^{-15}$.” This is shorthand for treating w, x, y, z as the amplitudes on $|1\rangle$ in each of the four single-qubit factors and suppressing the input dependence; the equations correctly identify the factorization above.

Example. For $x = 3 = (0011)_2$, the basis state $|0011\rangle$ should have amplitude ω^{-3x} . From the factorization, its amplitude is $1 \cdot 1 \cdot \omega^{-2x} \cdot \omega^{-x} = \omega^{-3x}$, as required.

Reducing each factor to known gates. We now express each $|\gamma_j\rangle$ as a sequence of standard gates applied to $|x_k\rangle$ for appropriate k .

Factor $|\gamma_3\rangle$. Since $\omega^{-8x} = (\omega^{-8})^x = (-1)^x$ and the parity of x equals x_0 (its least significant bit),

$$|\gamma_3\rangle = \frac{|0\rangle + (-1)^{x_0} |1\rangle}{\sqrt{2}} = H |x_0\rangle.$$

Factor $|\gamma_2\rangle$. We expand $\omega^{-4x} = \omega^{-4(8x_3+4x_2+2x_1+x_0)} = \omega^{-32x_3} \cdot \omega^{-16x_2} \cdot \omega^{-8x_1} \cdot \omega^{-4x_0} = (-1)^{x_1} (\omega^{-4})^{x_0}$, where the first two factors trivialize using $\omega^{16} = 1$. Hence

$$|\gamma_2\rangle = \frac{|0\rangle + (-1)^{x_1} (\omega^{-4})^{x_0} |1\rangle}{\sqrt{2}},$$

which can be obtained from $|x_1\rangle$ by applying a Hadamard gate followed by a controlled- ω^{-4} phase rotation controlled by $|x_0\rangle$.

Factors $|\gamma_1\rangle$ and $|\gamma_0\rangle$. By analogous expansions,

$$|\gamma_1\rangle = \frac{|0\rangle + (-1)^{x_2} (\omega^{-4})^{x_1} (\omega^{-2})^{x_0} |1\rangle}{\sqrt{2}},$$

$$|\gamma_0\rangle = \frac{|0\rangle + (-1)^{x_3} (\omega^{-4})^{x_2} (\omega^{-2})^{x_1} (\omega^{-1})^{x_0} |1\rangle}{\sqrt{2}}.$$

Thus $|\gamma_1\rangle$ is obtained from $|x_2\rangle$ by a Hadamard plus controlled- ω^{-4} and controlled- ω^{-2} phases (from $|x_1\rangle$ and $|x_0\rangle$ respectively), and $|\gamma_0\rangle$ is obtained from $|x_3\rangle$ by a Hadamard plus controlled- ω^{-4} , controlled- ω^{-2} , and controlled- ω^{-1} phases.

Combining these gives the circuit shown in Figure 14.

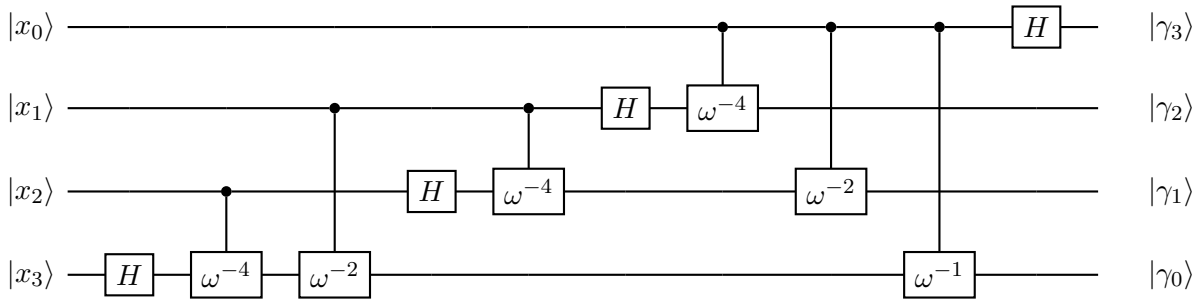


Figure 14: Circuit implementing the QFT for $N = 16$.

Remark. In matrix form,

$$\omega^{-4} = \begin{bmatrix} 1 & 0 \\ 0 & \omega^{-4} \end{bmatrix}, \quad \omega^{-2} = \begin{bmatrix} 1 & 0 \\ 0 & \omega^{-2} \end{bmatrix}, \quad \omega^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & \omega^{-1} \end{bmatrix}.$$

Remark. The same construction generalizes to $N = 2^n$ for any $n \in \mathbb{Z}^+$ and uses $O(n^2)$ gates. More generally, there is a circuit implementing the QFT over \mathbb{Z}_p^n :

$$\sum_{x \in [0, p-1]^n} \alpha_x |x\rangle \mapsto \sum_{\sigma, x \in [0, p-1]^n} \omega^{-\sigma \cdot x} \alpha_x |\sigma\rangle,$$

where $\sigma \cdot x = \sum_i \sigma_i x_i \pmod p$ and $\omega = e^{2\pi i/p}$.

8.2 Applications of the QFT

Example. Suppose we hold the state $\sum_{x \in [0, p-1]} |x\rangle |f(x)\rangle$ where $f(x) = ax + b \pmod p$ and the goal is to recover (a, b) . Applying the QFT over \mathbb{Z}_p^2 to *both* registers gives

$$\begin{aligned} \sum_x |x\rangle |ax + b\rangle &\mapsto \sum_{\sigma_1, \sigma_2} \sum_x \omega^{-(\sigma_1 x + \sigma_2(ax+b))} |\sigma_1 \sigma_2\rangle \\ &= \sum_{\sigma_1, \sigma_2} \sum_x \omega^{-x(\sigma_1 + \sigma_2 a)} \omega^{-\sigma_2 b} |\sigma_1 \sigma_2\rangle. \end{aligned}$$

The sum $\sum_x \omega^{-x(\sigma_1 + \sigma_2 a)}$ vanishes unless $\sigma_1 + \sigma_2 a \equiv 0 \pmod p$. Measuring therefore yields a uniformly random pair (σ_1, σ_2) satisfying this linear constraint. We leave it to the homework to show how this enables recovery of a and then b .

Example. Given the state $\sum_x \omega^{cx} |x\rangle$, applying the QFT yields

$$\sum_x \omega^{cx} |x\rangle \mapsto \sum_{\sigma} \left(\sum_x \omega^{(c-\sigma)x} \right) |\sigma\rangle.$$

The inner sum vanishes unless $c = \sigma$, so measurement returns c deterministically.

8.3 Mixed States

So far we have considered *pure states*, written as $|\psi\rangle = \sum_x \alpha_x |x\rangle$. Suppose, however, that we are presented with a quantum state sampled at random from some distribution over pure states: e.g., $|0\rangle$ and $|1\rangle$ each with probability $1/2$. Such a state cannot be written as any single pure state; we call it a *mixed state* and represent it as an *ensemble* $\{(p_j, |\psi_j\rangle)\}$.

Example. Sampling $|0\rangle$ or $|1\rangle$ each with probability $1/2$ gives the ensemble $\{(\frac{1}{2}, |0\rangle), (\frac{1}{2}, |1\rangle)\}$.

Example. Sampling $|+\rangle$ with probability 0.999 , and $|0\rangle, |1\rangle$ each with probability 0.0005 , gives the ensemble $\{(0.999, |+\rangle), (0.0005, |0\rangle), (0.0005, |1\rangle)\}$.

Density matrices. Any ensemble $\{(p_j, |\psi_j\rangle)\}$ can be represented by a *density matrix*

$$\rho = \sum_j p_j |\psi_j\rangle \langle \psi_j|.$$

Definition 8. A matrix ρ is a *density matrix* if it is

1. positive semi-definite: $\langle x | \rho | x \rangle \geq 0$ for all $|x\rangle \in \mathbb{C}^N$;
2. Hermitian: $\rho = \rho^\dagger$;
3. normalized: $\text{tr}(\rho) = 1$.

Pure states as density matrices. For a pure state $|\psi\rangle = \sum_i \alpha_i |i\rangle$, the density matrix is the outer product $\rho = |\psi\rangle \langle \psi|$, with entries $\rho_{ij} = \alpha_i \alpha_j^*$. Each of the three defining properties is easily verified:

- $\text{tr}(\rho) = \sum_i |\alpha_i|^2 = 1$.
- $\rho_{ji}^* = (\alpha_j \alpha_i^*)^* = \alpha_j^* \alpha_i = \rho_{ij}$, so $\rho = \rho^\dagger$.
- $\langle x | \rho | x \rangle = \langle x | \psi \rangle \langle \psi | x \rangle = |\langle x | \psi \rangle|^2 \geq 0$.

Theorem 8.1. For every ensemble $\{(p_j, |\psi_j\rangle)\}$, the matrix $\rho = \sum_j p_j |\psi_j\rangle \langle \psi_j|$ is a density matrix.

Proof. • $\text{tr}(\rho) = \sum_j p_j \text{tr}(|\psi_j\rangle \langle \psi_j|) = \sum_j p_j = 1$.

- ρ is a convex combination of Hermitian matrices, hence Hermitian.
- For any $|x\rangle$, $\langle x | \rho | x \rangle = \sum_j p_j |\langle x | \psi_j \rangle|^2 \geq 0$, so ρ is positive semi-definite.

□

Remark. The converse fails: distinct ensembles may give rise to the same density matrix.

Example. The ensembles

$$S_1 = \left\{ \left(\frac{3}{4}, |0\rangle \right), \left(\frac{1}{4}, |1\rangle \right) \right\} \quad \text{and} \quad S_2 = \left\{ \left(\frac{1}{2}, \frac{\sqrt{3}}{2} |0\rangle + \frac{1}{2} |1\rangle \right), \left(\frac{1}{2}, \frac{\sqrt{3}}{2} |0\rangle - \frac{1}{2} |1\rangle \right) \right\}$$

both give density matrix

$$\rho = \begin{bmatrix} \frac{3}{4} & 0 \\ 0 & \frac{1}{4} \end{bmatrix}.$$

We will see in the next lecture that no physical procedure can distinguish ensembles that share a density matrix; this fact will be foundational for quantum cryptography.

Key Takeaways. The QFT over \mathbb{Z}_{2^n} factors as a product of single-qubit states, yielding a circuit of $O(n^2)$ gates built from Hadamards and controlled phase rotations. A mixed state is a classical distribution over pure states, summarized by its density matrix $\rho = \sum_j p_j |\psi_j\rangle \langle \psi_j|$. Density matrices are positive semi-definite, Hermitian, and have trace 1; distinct ensembles can share a density matrix, foreshadowing the role of indistinguishability in quantum cryptography.

8.4 Further Reading

- Mixed states and measurement: Lecture 16 of [OW15].
- Quantum Fourier transform over \mathbb{Z}_N : Lecture 7 of [OW15].

9 Properties of Mixed States and Introduction to Quantum Key Distribution

In this lecture we develop more properties of mixed states and use them to introduce quantum key distribution (QKD). Recall that a *mixed state* is a classical probability distribution over pure states, written as an ensemble $\{(p_i, |\psi_i\rangle)\}_{i \in [K]}$: the state is $|\psi_i\rangle$ with probability p_i .

9.1 Measuring Pure and Mixed States

Measuring pure states. Consider a pure state $|\psi\rangle = \sum_i \alpha_i |i\rangle$ measured in the computational basis $\{|1\rangle, \dots, |N\rangle\}$. The outcome “ i ” occurs with probability

$$\Pr[\text{“}i\text{”}] = |\alpha_i|^2.$$

More generally, measuring $|\psi\rangle$ in an arbitrary orthonormal basis $\{|v_1\rangle, \dots, |v_N\rangle\}$ yields “ v_i ” with probability

$$\Pr[\text{“}v_i\text{”}] = |\langle v_i | \psi \rangle|^2.$$

Measuring mixed states. Now consider a mixed state $\{(p_j, |\psi_j\rangle)\}$ measured in the orthonormal basis $\{|v_1\rangle, \dots, |v_N\rangle\}$. By the law of total probability:

$$\begin{aligned} \Pr[\text{“}v_i\text{”}] &= \sum_j p_j |\langle v_i | \psi_j \rangle|^2 \\ &= \sum_j p_j \langle v_i | \psi_j \rangle \langle \psi_j | v_i \rangle \\ &= \langle v_i | \left(\sum_j p_j |\psi_j\rangle \langle \psi_j| \right) | v_i \rangle \\ &= \langle v_i | \rho | v_i \rangle, \end{aligned}$$

where we identified $\rho = \sum_j p_j |\psi_j\rangle \langle \psi_j|$ as the density matrix of the ensemble.

Remark. Any observable quantity of a mixed state depends only on its density matrix ρ . In particular, two ensembles with the same density matrix produce identical measurement statistics in every basis.

9.2 Properties of Density Matrices

We now formally verify the three defining properties of density matrices.

Property 1: trace one. Measuring ρ in the computational basis gives outcome “ i ” with probability

$$\Pr[“i”] = \langle i | \rho | i \rangle = \rho_{ii},$$

since $\langle i |$ picks out the i -th diagonal entry of ρ . Probabilities must sum to 1, so

$$\text{tr}(\rho) = \sum_i \rho_{ii} = \sum_i \Pr[“i”] = 1.$$

Density Matrix Property 1: $\text{tr}(\rho) = 1$.

Property 2: Hermitian. Writing $|\psi_i\rangle = \sum_j \alpha_j^i |j\rangle$,

$$\rho = \sum_i p_i |\psi_i\rangle \langle \psi_i|$$

has (j, k) -entry $\rho_{jk} = \sum_i p_i \alpha_j^i (\alpha_k^i)^*$. Its conjugate transpose has entry $\rho_{jk}^\dagger = \overline{\rho_{kj}} = \sum_i p_i (\alpha_k^i (\alpha_j^i)^*)^* = \sum_i p_i \alpha_j^i (\alpha_k^i)^* = \rho_{jk}$. Hence $\rho = \rho^\dagger$.

Density Matrix Property 2: ρ is Hermitian.

Property 3: positive semi-definite. For any $|v\rangle \in \mathbb{C}^N$ we must show $\langle v | \rho | v \rangle \geq 0$. Without loss of generality assume $|v\rangle$ is a unit vector and extend it to an orthonormal basis $\{|v_1\rangle = |v\rangle, |v_2\rangle, \dots, |v_N\rangle\}$. From the previous section, measuring ρ in this basis yields outcome “ v_i ” with probability $\langle v_i | \rho | v_i \rangle \geq 0$. In particular,

$$\langle v | \rho | v \rangle = \langle v_1 | \rho | v_1 \rangle = \Pr[“v_1”] \geq 0.$$

Density Matrix Property 3: $\rho \succeq 0$, i.e. $\langle v | \rho | v \rangle \geq 0$ for all $|v\rangle$.

9.3 Unitary Operations on Mixed States

Remark. Unitary operations on mixed states act on density matrices via $\rho \mapsto U\rho U^\dagger$.

To see this, let $S_1 = \{(p_i, |\psi_i\rangle)\}$ and define $S_2 = \{(p_i, U|\psi_i\rangle)\}$. Then

$$\begin{aligned} \rho_{S_2} &= \sum_i p_i U |\psi_i\rangle \langle \psi_i| U^\dagger = \sum_i p_i U |\psi_i\rangle \langle \psi_i| U^\dagger \\ &= U \left(\sum_i p_i |\psi_i\rangle \langle \psi_i| \right) U^\dagger = U \rho_{S_1} U^\dagger. \end{aligned}$$

Hence two ensembles with the same density matrix evolve to ensembles with the same density matrix under any fixed unitary.

9.4 Spectral Form of a Density Matrix

We now establish the converse: *every* positive semi-definite Hermitian matrix with trace 1 is the density matrix of some ensemble. Recall the spectral theorem for Hermitian matrices.

Theorem 9.1 (Spectral theorem). *For every Hermitian matrix M acting on \mathbb{C}^N , there exists an orthonormal basis $\{|v_1\rangle, \dots, |v_N\rangle\}$ of \mathbb{C}^N and real eigenvalues $\lambda_1, \dots, \lambda_N$ such that*

$$M = \sum_j \lambda_j |v_j\rangle \langle v_j|, \quad M |v_j\rangle = \lambda_j |v_j\rangle.$$

Given a density matrix ρ with spectral decomposition $\rho = \sum_j \lambda_j |v_j\rangle \langle v_j|$, we now show $\lambda_j \geq 0$ for all j and $\sum_j \lambda_j = 1$.

For any orthonormal basis $\{|v_1\rangle, \dots, |v_N\rangle\}$, there exists a unitary U such that $U |v_i\rangle = |i\rangle$ and $U^\dagger |i\rangle = |v_i\rangle$. Applying U to ρ yields $\rho' = U\rho U^\dagger$, which by direct calculation has $\langle i | \rho' | i \rangle = \langle v_i | \rho | v_i \rangle = \lambda_i$. Thus

$$\rho' = \sum_i \lambda_i |i\rangle \langle i| = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_N \end{bmatrix}.$$

Since $\rho \succeq 0$ implies $\rho' \succeq 0$, each $\lambda_i \geq 0$. Since $\text{tr}(\rho') = \text{tr}(\rho) = 1$, we get $\sum_i \lambda_i = 1$.

Theorem 9.2 (Canonical decomposition). *Every density matrix ρ with distinct eigenvalues $\lambda_1, \dots, \lambda_N$ corresponds to the canonical mixed-state ensemble $\{(\lambda_i, |v_i\rangle)\}_{i \in [N]}$, where $\{|v_i\rangle\}$ is the eigenbasis of ρ .*

Remark. When eigenvalues are degenerate, multiple distinct canonical decompositions exist, since any orthonormal basis of the relevant eigenspace yields an equally valid ensemble. A special case, where *all* eigenvalues coincide, gives rise to the *maximally mixed state*, central to many cryptographic constructions.

9.5 The Maximally Mixed State

A density matrix is *maximally mixed* if all its eigenvalues are equal. Since the eigenvalues sum to 1, each must equal $1/N$ where N is the dimension. Thus

$$\rho = \frac{1}{N} \mathbb{I} = \begin{bmatrix} 1/N & & \\ & 1/N & \\ & & \ddots \\ & & & 1/N \end{bmatrix}.$$

The maximally mixed state is the same in every basis, which is the key reason it is so useful in cryptography.

Example (Game 1). Consider the ensemble $\{(\frac{1}{2}, |0\rangle), (\frac{1}{2}, |1\rangle)\}$. Its density matrix is

$$\begin{aligned} \rho &= \frac{1}{2} |0\rangle \langle 0| + \frac{1}{2} |1\rangle \langle 1| \\ &= \frac{1}{2} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix}, \end{aligned}$$

which is the maximally mixed state on one qubit.

Example (Game 2). Consider the ensemble $\{(\frac{1}{2}, |+\rangle), (\frac{1}{2}, |-\rangle)\}$. Its density matrix is

$$\begin{aligned}\rho &= \frac{1}{2} |+\rangle \langle +| + \frac{1}{2} |-\rangle \langle -| \\ &= \frac{1}{2} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix},\end{aligned}$$

which agrees with Game 1.

Since both ensembles share a density matrix, no measurement can distinguish them. This indistinguishability is what we will exploit in QKD: an eavesdropper who only sees a maximally mixed state has no information about the underlying ensemble.

9.6 Quantum Key Distribution: Setup

The idea of quantum key distribution originates with Stephen Wiesner, Charles Bennett, and Gilles Brassard.

Scenario. Two honest parties, Alice and Bob, wish to communicate over a classical (public) channel:

$$\text{Alice} \xleftarrow[\text{channel}]{\text{classical}} \text{Bob}.$$

An adversary Eve has access to the same classical channel and may observe or block messages, though not modify them or inject her own (the channel is *authenticated*). The goal is for Alice and Bob to agree on a key k that remains hidden from Eve in an *information-theoretic* sense, i.e. even against a computationally unbounded adversary.

Remark. If all three parties are computationally unbounded and only the classical channel is available, no such key-agreement protocol exists [IR89].

Now suppose Alice and Bob additionally have access to a *quantum channel*, which we model as *unauthenticated*: Eve can intercept, modify, replace, or drop quantum messages without being detected by the channel itself. Can the parties still agree on a key that Eve learns nothing about?

Options.

- *Classical channel only.* Reduces to the no-quantum-resource case, which is impossible by the above remark.
- *Quantum channel only.* Eve can mount a *man-in-the-middle attack*: she impersonates “Bob” to Alice and “Alice” to Bob, agreeing on separate keys with each. Neither party detects the deception.
- *Both channels.* Alice can sample n states uniformly at random from $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$ and send them over the quantum channel. Bob measures each in either the computational or Hadamard basis. They later use the authenticated classical channel to compare bases and check for tampering.

We will develop the third option (the BB84 protocol) in the next lecture.

Key Takeaways. Measurement outcomes depend only on the density matrix $\rho = \sum_j p_j |\psi_j\rangle \langle \psi_j|$, which is positive semi-definite, Hermitian, and trace 1; every such matrix is the density matrix of some ensemble. The maximally mixed state $\frac{1}{N}\mathbb{I}$ looks the same in every basis, which is why ensembles such as $\{(\frac{1}{2}, |0\rangle), (\frac{1}{2}, |1\rangle)\}$ and $\{(\frac{1}{2}, |+\rangle), (\frac{1}{2}, |-\rangle)\}$ are indistinguishable to an observer. Information-theoretic key agreement is impossible with only a classical channel, but a quantum channel offers a route around this, as we will see next lecture.

9.7 Further Reading

- Mixed states and measurement: Lecture 16 of [OW15].
- Quantum information theory and Holevo's bound: Lecture 18 of [OW15].

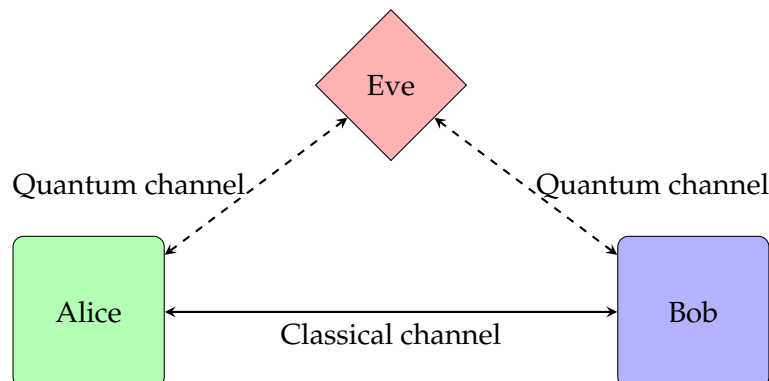
10 Quantum Key Distribution

In this lecture we present a cryptographic application of the quantum theory developed so far: a specific protocol for Quantum Key Distribution (QKD) [BF12]. The construction uses tools from earlier lectures — conjugate coding, the Hadamard basis, and a tractable adversarial model based on mixed states.

10.1 Problem Setup

Two honest parties, Alice and Bob, wish to agree on a shared random bitstring (the *key*). They have access to two communication channels:

- A *classical authenticated* channel: an adversary Eve can read or block messages, but cannot tamper with their contents or inject her own.
- A *quantum* channel: Eve has full control, including measuring, replacing, dropping, or otherwise modifying any quantum signal in transit.



The protocol has two goals:

1. *Agreement*. Alice and Bob end with the same key.
2. *Secrecy*. Eve obtains essentially no information about the key.

Eve can always cause the protocol to abort by dropping every message; what we want to rule out is that Alice and Bob accept *different* keys, or that Eve also knows the agreed-upon key.

10.2 Indistinguishability

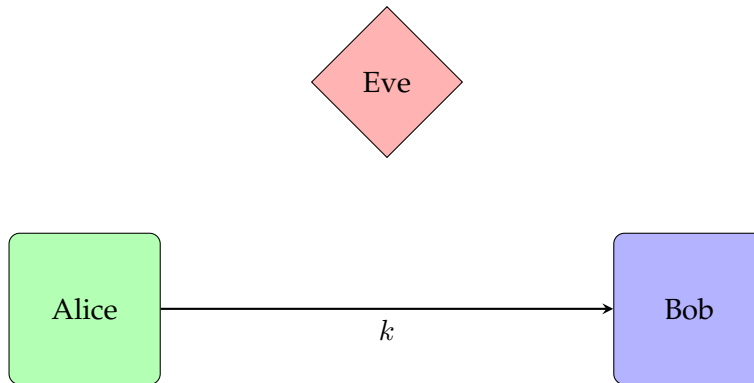
We will repeatedly say two distributions are “indistinguishable,” denoted \approx . Three flavors are relevant.

- **Perfect indistinguishability.** The distributions are identical.
- **Statistical (ϵ -) indistinguishability (\approx_ϵ).** The total variation distance is at most ϵ .
- **Computational indistinguishability.** No polynomial-time machine can distinguish the distributions except with negligible probability.

This lecture focuses on statistical indistinguishability.

10.3 Attempt 0: Classical Channel Only

Alice samples $k \xleftarrow{\$} \{0, 1\}^n$ and sends it to Bob over the classical channel.



This achieves agreement but fails secrecy: Eve observes the classical channel and learns k . Real-world classical key exchange (e.g., Diffie-Hellman) relies on additional cryptographic assumptions to hide k from computationally bounded adversaries; against an unbounded adversary it is impossible [ABB⁺14].

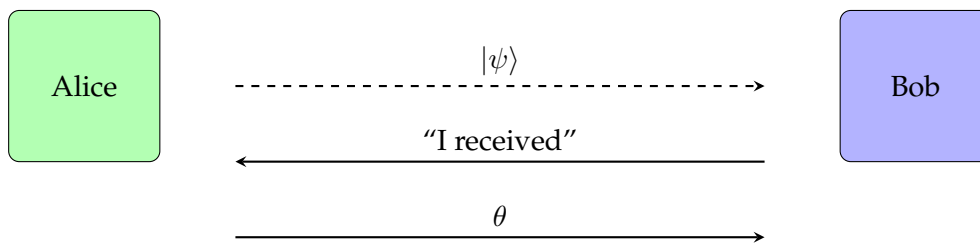
10.4 Attempt 1: Encoding in Random Bases

We introduce conjugate-coding notation. For $x, \theta \in \{0, 1\}$, write $|x\rangle_\theta$ for the basis state in basis θ :

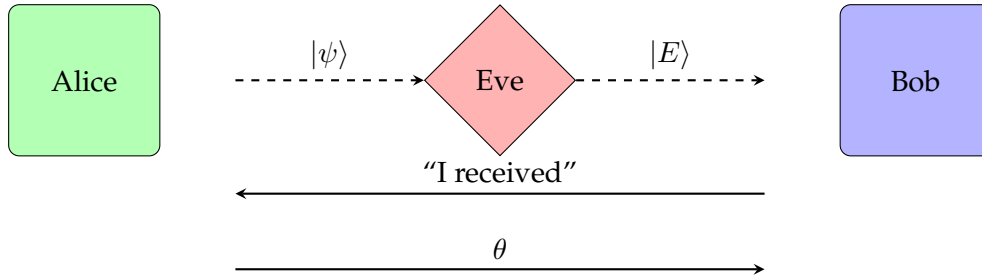
		θ	
		0	1
x	0	$ 0\rangle$	$ +\rangle$
	1	$ 1\rangle$	$ -\rangle$

Thus $\theta = 0$ is the computational basis and $\theta = 1$ is the Hadamard basis.

Alice samples $x, \theta \xleftarrow{\$} \{0, 1\}^n$ and prepares $|\psi\rangle = \bigotimes_i |x_i\rangle_{\theta_i}$. She sends $|\psi\rangle$ to Bob over the quantum channel. Bob acknowledges receipt; Alice then sends θ over the classical channel, and Bob measures each qubit in basis θ_i , recovering x_i exactly.



Why this fails against Eve. Eve sits on the quantum channel and is free to replace $|\psi\rangle$ with her own state $|E\rangle$:



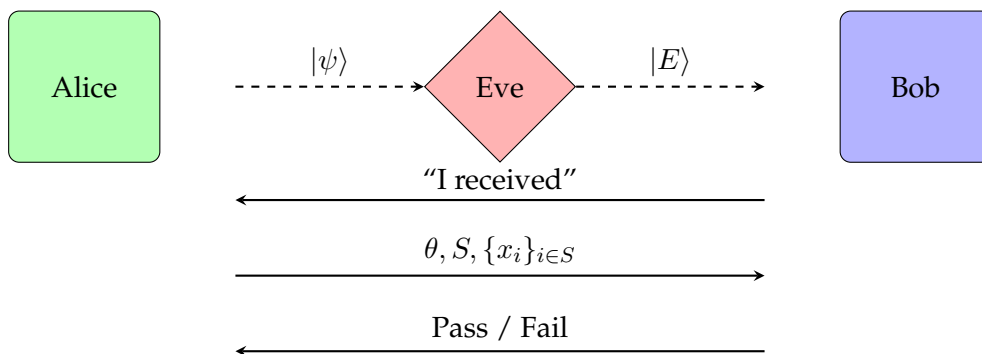
When Alice later announces θ , Eve also receives it. She can use θ to measure her copy of $|\psi\rangle$ correctly and recover x . Bob, meanwhile, decodes $|E\rangle$ in basis θ . Since Eve constructed $|E\rangle$ without knowing θ , half her bits (on average) are encoded in the opposite basis from what Bob measures, so Bob's measurements yield a random bit on those positions. Result: Alice and Eve share x ; Alice and Bob disagree on roughly half their bits. Both goals fail.

10.5 Attempt 2: Test Step

The remedy is to have Alice *test* that Bob's state is undisturbed before revealing θ in full.

Protocol.

1. Alice samples $x, \theta \xleftarrow{\$} \{0, 1\}^n$ and sends $|\psi\rangle = \bigotimes_i |x_i\rangle_{\theta_i}$.
2. Bob acknowledges receipt.
3. Alice samples a subset $S \subseteq [n]$ uniformly at random (the *test set*) and sends $(\theta, S, \{x_i\}_{i \in S})$.
4. Bob measures each received qubit $|\psi'_i\rangle$ in basis θ_i , obtaining y_i . He checks whether $y_i = x_i$ for all $i \in S$, reporting Pass or Fail to Alice.
5. On Pass, the parties output $\{x_i\}_{i \notin S}$ (Alice) and $\{y_i\}_{i \notin S}$ (Bob) as the key.



Intuition for why this works. Crucially, no quantum messages travel *after* the parameters S and θ are revealed, so Eve must commit to her tampering of $|\psi\rangle$ before learning either. If Eve replaces only a small number of qubits, the test will likely pass; but those qubits she replaced reveal nothing about x , since she chose them without knowing θ . If she tampers with many qubits, she is likely caught.

The size of S governs the tradeoff: larger S catches more tampering but uses up more bits, leaving a shorter key.

Noise. The protocol assumes a noiseless channel. In practice, channel noise is indistinguishable from low-rate adversarial tampering; we will need error correction to address this, and we do so in a later lecture.

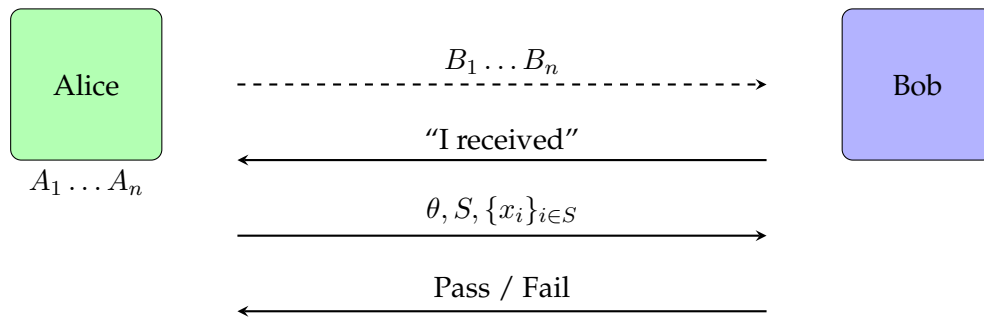
Removing the quantum memory requirement. The protocol as stated requires Bob to store $|\psi\rangle$ until θ arrives, which demands a high-quality quantum memory. A standard workaround is to have Bob measure each qubit immediately in a random basis $\hat{\theta}_i \xleftarrow{\$} \{0, 1\}$, then later restrict the test to indices where $\hat{\theta}_i = \theta_i$.

10.6 An Alternative Perspective: EPR Pairs

The protocol admits an equivalent reformulation that is conceptually cleaner for security analysis. Instead of sampling x and preparing $|x\rangle_\theta$, Alice prepares n EPR pairs

$$|A_i B_i\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

and sends the B_i halves to Bob.

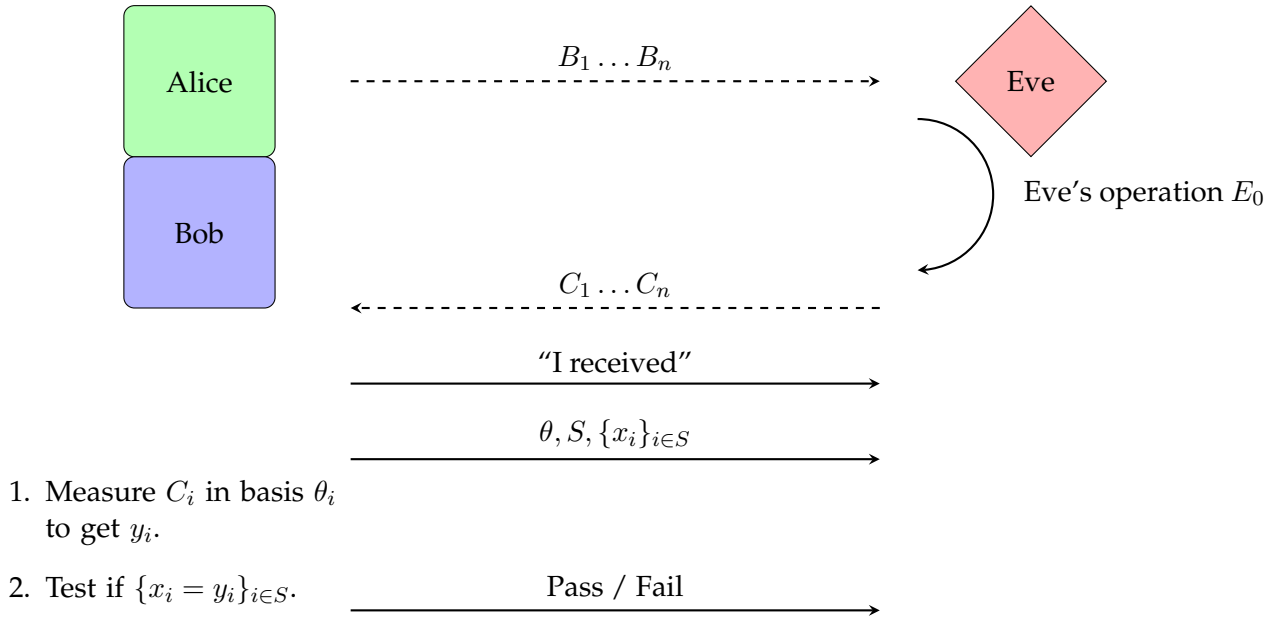


After Bob acknowledges, Alice measures each A_i in basis θ_i , which collapses B_i into the same basis state $|x_i\rangle_{\theta_i}$. The rest of the protocol proceeds as before.

Why this is equivalent. For each qubit, the marginal density matrix on Bob's side is $\frac{\mathbb{I}}{2}$ (the maximally mixed state), exactly as in the original protocol where Alice picked a random pure state in one of two random bases. The two protocols are therefore indistinguishable to Eve. They are equivalent for Alice's purposes too — Alice just commutes the order of "pick x_i " and "send the corresponding qubit."

10.7 The Adversarial View

To analyze secrecy, we group Alice and Bob together into a single entity and consider Eve's view.

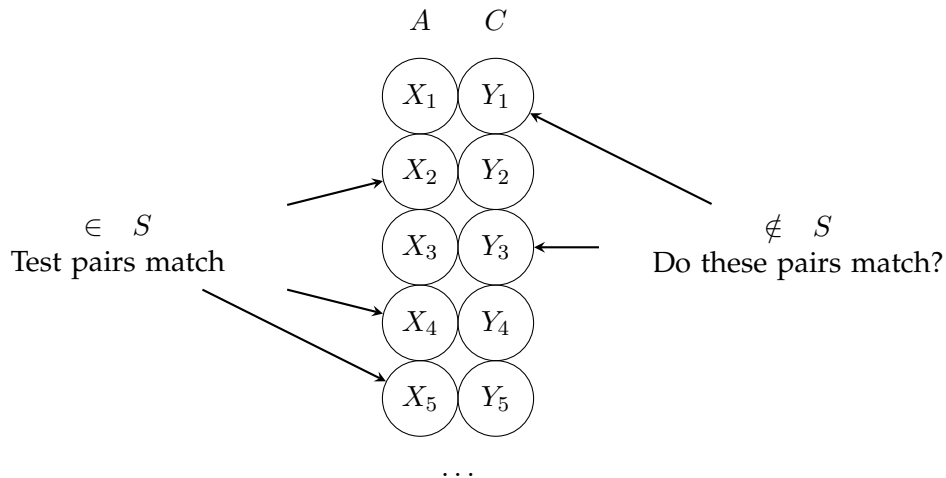


Eve receives B_1, \dots, B_n from Alice, applies some quantum operation E_0 (which may include adding ancilla qubits and entangling everything), and returns C_1, \dots, C_n to Bob. After this point, Eve passively observes the classical messages.

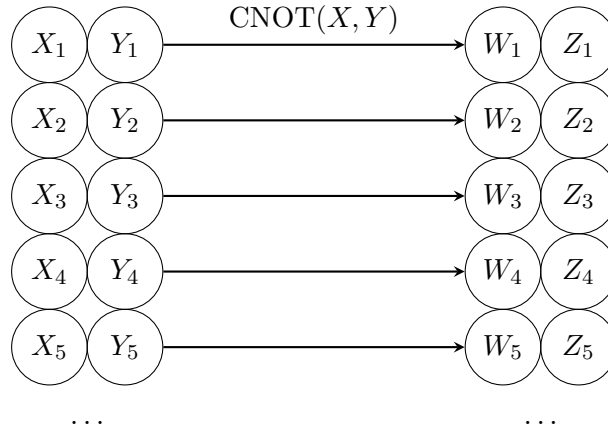
The overall procedure can now be abstracted as a *sampling experiment*: after measuring y_i from C_i in basis θ_i , the experiment reports Pass iff $x_i = y_i$ for all $i \in S$, in which case it outputs $\{x_i\}_{i \notin S}$ (Alice's key) and $\{y_i\}_{i \notin S}$ (Bob's key).

10.8 Sampling Game

We can repack the analysis visually. At the end of the protocol, the pairs (A_i, C_i) are measured, producing classical bit pairs (X_i, Y_i) . Pairs in S are tested; pairs outside S form the candidate key.

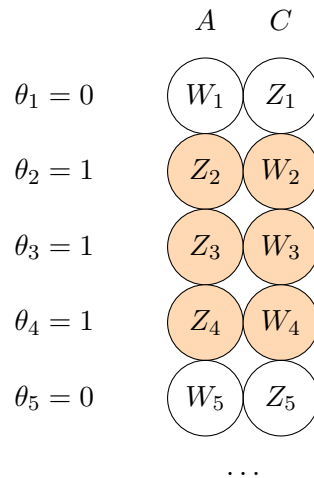


CNOT reformulation. Applying a CNOT between each X_i and Y_i (the latter as target) transforms each pair (X_i, Y_i) into $(W_i, Z_i) = (X_i, X_i \oplus Y_i)$. Since CNOT is reversible, no information is lost. The original test “ $X_i = Y_i$?” becomes “ $Z_i = 0$?”



This reframes the analysis: instead of bounding “how many bits do Alice and Bob disagree on,” we bound “how many Z_i are nonzero.”

Basis-dependent swap. A subtlety arises if the CNOT is applied *before* measurement: in the Hadamard basis (where $\theta_i = 1$), the CNOT propagates differently. Direct computation shows that the resulting roles of W_i and Z_i swap on indices with $\theta_i = 1$:



Computing the swap. The swap follows from the identity (for a single pair (x, y)):

$$\begin{aligned} H^{\otimes 2}(\text{CNOT}(H^{\otimes 2}(|x\rangle|y\rangle))) &= H^{\otimes 2}\left(\text{CNOT} \sum_{\sigma_1, \sigma_2} (-1)^{\sigma_1 x + \sigma_2 y} |\sigma_1 \sigma_2\rangle\right) \\ &= H^{\otimes 2} \sum_{\sigma_1, \sigma_2} (-1)^{\sigma_1 x + \sigma_2 y} |\sigma_1, \sigma_1 \oplus \sigma_2\rangle. \end{aligned}$$

Substituting $\sigma_3 = \sigma_1 \oplus \sigma_2$, and applying the final Hadamards, leads after some algebra to the state $|x \oplus y, y\rangle$. The original computational-basis CNOT mapped $(x, y) \rightarrow (x, x \oplus y) = (W, Z)$; in the Hadamard basis it instead maps $(x, y) \rightarrow (x \oplus y, y) = (Z, W)$, hence the swap.

10.9 Putting It Together

If the test “ $Z_i = 0$ for all $i \in S$ ” passes, Alice and Bob output $\{W_i\}_{i \notin S}$ as the shared key.⁷

In future lectures we make rigorous the statistical guarantees one obtains from the test — both about agreement (how many of the untested Z_i are also zero, hence how many key bits agree) and about secrecy (how much Eve can know about W_i for $i \notin S$).

Key Takeaways. The BB84-style QKD protocol uses Alice’s secret choice of basis to defeat man-in-the-middle attacks: Eve must commit to her tampering before learning θ , and any non-trivial tampering is detected with high probability via Alice’s random test set. The protocol can be reformulated using EPR pairs (Alice prepares pairs and measures her half) without changing what Eve sees, but the EPR picture makes the security analysis cleaner. Finally, applying a CNOT to each (X_i, Y_i) pair turns the question “do Alice and Bob agree?” into “how many Z_i are nonzero?”—a sampling problem we will analyze quantitatively next time.

⁷Here we are continuing to use the variable names W, Z assigned by basis; in implementation terms, W_i is Alice’s measured bit X_i if $\theta_i = 0$ and the parity $X_i \oplus Y_i$ if $\theta_i = 1$. Either way, both parties output the same value modulo statistical errors caught by the next lecture’s analysis.

11 Quantum Key Distribution – II

We continue our analysis of Quantum Key Distribution. Recall the setup: Alice and Bob aim to agree on a uniformly random key, even in the presence of an adversary Eve who controls the quantum channel. The protocol (as developed in the previous lecture) proceeds as follows.

1. Alice prepares n EPR pairs and sends Bob's halves B_1, \dots, B_n to Eve over the quantum channel.
2. Eve performs an arbitrary quantum operation, returning registers C_1, \dots, C_n to Bob.
3. Alice measures her halves A_1, \dots, A_n in bases $\theta_1, \dots, \theta_n$ to obtain X_1, \dots, X_n . Bob measures C_1, \dots, C_n in the same bases (after they are revealed) to obtain Y_1, \dots, Y_n .
4. Alice and Bob test $\{X_i = Y_i\}_{i \in S}$ on a random subset $S \subseteq [n]$. If the test passes, Alice's key is $\{X_i\}_{i \notin S}$ and Bob's key is $\{Y_i\}_{i \notin S}$.

Recall also from Section 10.8 the equivalent reformulation in (W, Z) variables: after CNOTing each pair, the test reduces to " $Z_i = 0$ for all $i \in S$." Concretely:

- Sample $\{\theta_i\}_{i \in [n]}$.
- Sample $S \subset [n]$ of size $n/2$.
- Measure $\{Z_i\}_{i \in S}$ and test if they are all zero.

If the test passes, we want to argue **agreement** (Alice's and Bob's keys are equal) and **secrecy** (Eve learns essentially nothing). We address these in turn.

11.1 Agreement: Intuition

Arrange the q_i 's in a $n \times 2$ matrix

$$\begin{pmatrix} q_1^0 & q_1^1 \\ q_2^0 & q_2^1 \\ \vdots & \vdots \\ q_n^0 & q_n^1 \end{pmatrix}$$

where the first column represents the A register (Hadamard basis, abbreviated H) and the second column the C register (computational basis, abbreviated C). The agreement argument proceeds in three steps:

1. Sample $\theta_i \xleftarrow{\$} \{C, H\}$ for each $i \in [n]$, and set $j_i = 0$ if $\theta_i = H$, $j_i = 1$ if $\theta_i = C$. Let $T = \{(i, j_i)\}_{i \in [n]}$ (the circled positions above).

2. Sample $S \subset T$ with $|S| = n/2$ (the circled *and* highlighted positions) and measure q_S to obtain Z_S . The plan is to use Z_S to estimate $Z_{T \setminus S}$.
3. When Z_S are all zero, we will argue that, except with probability ε , the registers $q_{T \setminus S}$ behave like a superposition $\sum_{u: w(u) \leq \delta n} \alpha_u |u\rangle$ supported on low-Hamming-weight strings, where ε is the quantum sampling error.

11.2 Secrecy: Intuition

Recall that the registers in T produce the W_i 's and those in \bar{T} produce the Z_i 's, with W_i obtained by measuring in the conjugate basis to the one used for Z_i . If Z_S are all zero, then:

- Most left (A) registers are close to $|+\rangle$.
- Most right (C) registers are close to $|0\rangle$.

We will show that under these conditions, the non-circled registers must also yield zeros (in the Hadamard basis on the left and the computational basis on the right), which forces Eve's knowledge of the resulting key to be limited.

11.3 Classical Sampling and Estimation

We first analyze a purely classical analogue. Suppose we hold a $2n$ -bit string, represented as an $n \times 2$ matrix

$$\begin{pmatrix} q_1^0 & q_1^1 \\ q_2^0 & q_2^1 \\ \vdots & \vdots \\ q_n^0 & q_n^1 \end{pmatrix}.$$

The goal is to sample a few bits in accordance with a specified sampling strategy and estimate the Hamming weight of the entire string.

For each $i \in [n]$, sample $j_i \stackrel{\$}{\leftarrow} \{0, 1\}$ and let $T = \{(i, j_i)\}_{i \in [n]}$ and $\bar{T} = \{(i, 1 - j_i)\}_{i \in [n]}$. Next sample $S \subset T$ with $|S| = n/2$ and observe $w[q_S]$ (the Hamming weight of q_S). We aim to use $w[q_S]$ to estimate both

$$w[q_{T \setminus S}] = w[q_T] - w[q_S] \quad (\text{for agreement})$$

and

$$w[q_{\bar{T}}] \quad (\text{for privacy, by argument later in this section}).$$

We split the analysis into three steps.

11.3.1 Step 1: Relating $w[q_T]$ and $w[q_S]$

By Hoeffding's inequality, since q_S is obtained by sampling $|S| = n/2$ positions of q_T without replacement,

$$\Pr \left[\left| \frac{w[q_S]}{|S|} - \frac{w[q_T]}{|T|} \right| \geq \delta \right] \leq 2e^{-2\delta^2|S|}. \quad (11.1)$$

With $|S| = n/2$ and $|T| = n$,

$$\Pr \left[|2w[q_S] - w[q_T]| \geq \delta n \right] \leq 2e^{-\delta^2 n}. \quad (11.2)$$

In particular, conditioned on $w[q_S] = 0$,

$$\Pr[w[q_T] \leq \delta n] \geq 1 - 2e^{-\delta^2 n}. \quad (11.3)$$

This is the classical analogue of the agreement game. By the Bouman–Fehr theorem [BF12], the quantum error satisfies $\varepsilon_q \leq \sqrt{\varepsilon_{\text{cl}}} \leq \sqrt{2} e^{-\delta^2 n/2}$.

11.3.2 Step 2: Relating $w[q_T]$ and $w[q_{\overline{T}}]$

Now consider the privacy-side estimate. Let

$$L = \{i : q_i^{j_i} \neq q_i^{1-j_i}\}, \quad \ell = |L|.$$

On indices outside L , $q_i^{j_i} = q_i^{1-j_i}$ contributes equally to $w[q_T]$ and $w[q_{\overline{T}}]$ and cancels in the difference. On indices in L , exactly one of $q_i^{j_i}, q_i^{1-j_i}$ is 1. Therefore

$$w[q_T] - w[q_{\overline{T}}] = w[q_{T|L}] - w[q_{\overline{T}|L}] = 2w[q_{T|L}] - \ell.$$

Now $w[q_{T|L}]$ has mean $\ell/2$ (since for each $i \in L$, $q_i^{j_i}$ is a uniformly random one of two complementary bits). Hoeffding gives, for $|X_i| \leq 1$ and $S = X_1 + \dots + X_m$,

$$\Pr[|S - \mathbb{E}[S]| \geq t] \leq 2e^{-2t^2/m}.$$

Substituting $X_i = q_i^{j_i}$ for $i \in L$, $m = \ell \leq n$, and $t = \varepsilon n/2$,

$$\Pr[|w[q_T] - w[q_{\overline{T}}]| \geq \varepsilon n] = \Pr[|w[q_{T|L}] - \ell/2| \geq \varepsilon n/2] \leq 2e^{-n\varepsilon^2/2}. \quad (11.4)$$

11.3.3 Step 3: Combining Steps 1 and 2

From (11.1) and the rescaled (11.4) we obtain, via the triangle inequality,

$$\Pr\left[\left|\frac{w[q_S]}{|S|} - \frac{w[q_{\overline{T}}]}{|\overline{T}|}\right| \geq \delta + \varepsilon\right] \leq 2e^{-n\varepsilon^2/2} + 2e^{-\delta^2 n}. \quad (11.5)$$

Setting $\varepsilon = \delta$,

$$\Pr\left[\left|\frac{w[q_S]}{|S|} - \frac{w[q_{\overline{T}}]}{|\overline{T}|}\right| \geq 2\delta\right] \leq 4e^{-\delta^2 n}. \quad (11.6)$$

In the regime $w[q_S] = 0$ and $\varepsilon = \delta = 10^{-3}$,

$$\Pr[w[q_{\overline{T}}] \geq 0.002n] \leq 4e^{-n/(2 \times 10^6)}. \quad (11.7)$$

11.3.4 Converting Classical to Quantum Bounds

By [BF12], the quantum sampling error is bounded by the square root of the classical error:

$$\varepsilon_q \leq \sqrt{\varepsilon_{\text{cl}}} = \sqrt{4e^{-n/(2 \times 10^6)}} = 2e^{-n/(4 \times 10^6)}. \quad (11.8)$$

Returning to the quantum game, except with probability at most $2e^{-n/(4 \times 10^6)}$, the registers $q_{T \setminus S}$ are ε_q -close to a state of the form

$$\sum_{T,S} |T, S\rangle \langle T, S| \otimes \sum_{i \in \{0,1\}^n} \alpha_i |i\rangle |\phi_E^i\rangle, \quad (11.9)$$

where $\alpha_i = 0$ for all i with Hamming weight greater than δn , and $|\phi_E^i\rangle$ are arbitrary states held by Eve. Thus, conditioned on the test passing, the post-protocol state lies in the low-Hamming-weight subspace, exactly as the intuition suggested.

Key Takeaways. The QKD test is, in essence, a sampling experiment: by checking a random half of the qubit pairs, Alice and Bob obtain high-confidence estimates for the entire bitstring. Hoeffding's inequality bounds the classical version of this estimate; the Bouman–Fehr theorem [BF12] bridges from classical to quantum sampling errors at the cost of a square root. The upshot is that conditional on the test passing, the residual state of Alice's and Bob's qubits is, up to a negligible error, a superposition of low-Hamming-weight strings — the foundation of the next lecture's secrecy argument.

11.4 Further Reading

- Niek J. Bouman and Serge Fehr. *Sampling in a Quantum Population, and Applications*. arXiv:0907.4246, 2012.
- Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2016.

12 Finishing QKD and Introduction to Quantum Oblivious Transfer

In this lecture we complete our account of quantum key distribution following the security proof of [BF12]. Specifically, we show how the “weak” agreement and “weak” privacy claims established in the previous lecture can be strengthened using error correction and privacy amplification, respectively. We then introduce *oblivious transfer* (OT) and discuss its information-theoretic impossibility, motivating the use of additional cryptographic primitives.

12.1 Error Correction in QKD

Recall that in the previous lecture we established *weak* agreement: if Alice and Bob find that the bits sampled on $S \subseteq T$ (with $|S| \leq n/2$) are all zero, then the untested portion of the resulting bitstring is approximately a superposition of strings of low Hamming weight $w(s) \leq \delta n$, with probability of failure exponentially small in n .

This is not yet sufficient: even a small Hamming-weight discrepancy means Alice’s and Bob’s strings disagree on a few positions, so if Alice encrypts a message by XORing it with her string, Bob’s decryption produces gibberish on those positions. We need a method that yields *exact* agreement. The standard tool is an error-correcting code, defined as follows.

Definition 9. Let $\delta \in [0, 1]$ and $k, n \in \mathbb{N}$. A δn -error-correcting code $\text{ECC}_{\delta n} = (\text{Enc}, \text{Dec})$ consists of an encoding function $\text{Enc}: \{0, 1\}^k \rightarrow \{0, 1\}^n$ and a decoding function $\text{Dec}: \{0, 1\}^n \rightarrow \{0, 1\}^k$ such that for all $m \in \{0, 1\}^k$, $\text{Dec}(y) = m$ whenever $w(y \oplus \text{Enc}(m)) \leq \delta n$.

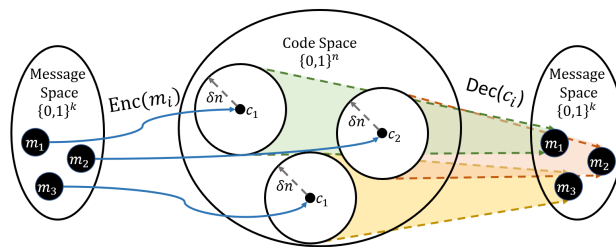


Figure 15: An error-correcting code: each message has a “codespace” that absorbs up to δn errors.

The intuition (Figure 15) is that n is large compared to k , so each codeword has a δn -radius ball around it that contains no other codeword. We refer the reader to [MS77] for the existence and construction of such codes.

QKD on top of an ECC. We modify the QKD protocol as follows (see Figure 16):

1. Alice and Bob fix a δn -error-correcting code ECC in advance.

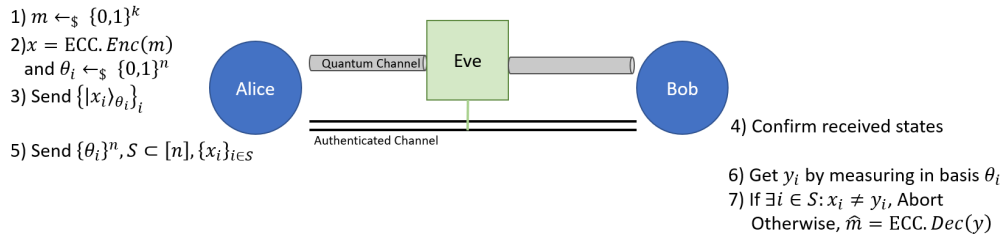


Figure 16: QKD implemented on top of an error-correcting code.

2. Alice samples $m \leftarrow_{\mathcal{S}} \{0, 1\}^k$ uniformly.
3. Alice computes $c = \text{ECC.Enc}(m)$.
4. Alice sets $x_i = c_i$ for $i \in [n]$.
5. Alice and Bob run the standard QKD protocol so that Bob obtains $y \in \{0, 1\}^n$ with $w(y \oplus x) \leq \delta n$ (with high probability, conditional on the sampling test passing).
6. Bob computes $\hat{m} = \text{ECC.Dec}(y)$, recovering m with high probability.

Subtlety: x is no longer uniform. Because $x = \text{Enc}(m)$ is a codeword rather than a uniformly random string, the original assumption of uniform x in our security analysis no longer holds directly. However, the Bouman–Fehr sampling argument was *sampling-without-replacement* based, hence applies regardless of whether x is uniform; Hoeffding’s inequality continues to bound the error. Informally, the randomness now flows from m and the basis information (used as tamper detection on the codeword), rather than from x directly.

Alternative: syndrome-based error correction. A common variant, used in [BF12] and many subsequent works, is for Alice to send Bob a small *syndrome* s_x of x over the classical channel. The syndrome is a compact summary that lets Bob correct y to x given y ; it is chosen to leak as little as possible about x to Eve.

12.2 Privacy in QKD

We have only argued *weak* privacy: the state at the end of QKD is close to a superposition of low-Hamming-weight strings. We now connect this to a quantitative notion of privacy.

At the end of the protocol, Alice and Bob hold the same $x \in \{0, 1\}^k$ with high probability, and Eve holds a quantum register E (her “view”), which encapsulates everything she has learned — the modified quantum states, the classical communication, and any side information she may have. The question is: how well can Eve guess $\hat{x} = x$?

It is a classical result of [KRS09] that Eve’s optimal guessing strategy succeeds with probability

$$\Pr[\hat{x} = x] = 2^{-H_{\min}(X|E)_{\rho_{XE}}},$$

where ρ_{XE} is the joint state of x and Eve’s register and H_{\min} is the *conditional min-entropy*. If $H_{\min}(X | E)$ scales linearly in n , Eve’s success probability decays exponentially in n . The Bouman–Fehr analysis [BF12] establishes precisely such a bound: the low-Hamming-weight superposition

state from the previous lecture has $H_{\min}(X | E) \approx cn$ for some constant $c \in (0, 1)$ (close to 1 under ideal parameter settings). This is what we mean by *weak* privacy.

From weak privacy to strong privacy. For *strong* privacy, we want a key $\ell \in \{0, 1\}^{k'}$ such that

$$\Pr[\widehat{\ell} = \ell] \approx 2^{-k'},$$

i.e. Eve's guess is no better than uniform. This is achieved by applying a quantum-safe *extractor* to x : a function $\text{Ext}(x, r)$, where r is a public random seed, that outputs a uniformly random $\ell \in \{0, 1\}^{k'}$ as long as the underlying X has enough min-entropy. The standard bound requires $2k' \lesssim H_{\min}(X | E)$.⁸

The bound on H_{\min} from the previous step thus determines the achievable key length k' . Construction and analysis of quantum-safe extractors are beyond our scope; see [Ren05], Chapter 5.

12.3 Oblivious Transfer

We now turn to a new task: *oblivious transfer* (OT).

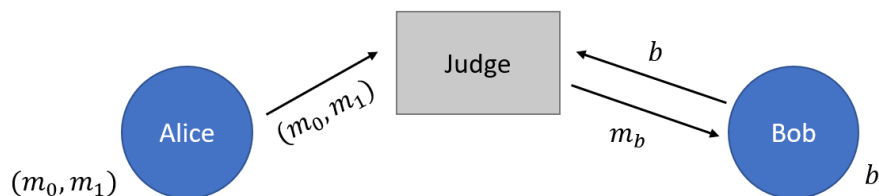


Figure 17: Ideal functionality of 1-out-of-2 oblivious transfer.

Alice holds two bits m_0, m_1 ; Bob holds a choice bit b . The goal is for Bob to learn m_b (and nothing about m_{1-b}), and for Alice to learn nothing about b . Formally:

1. Alice does not learn b .
2. Bob does not learn m_{1-b} .

This is a form of private computation of the two-input function $f(b) := m_b$, where Alice never sees the output and Bob sees nothing beyond it. There is no eavesdropper here; both parties may be malicious. In the ideal functionality (Figure 17), a trusted “judge” takes both parties’ inputs and gives Bob the appropriate output. Our goal is to implement OT *without* the judge.

12.4 Security Against Malicious Senders

Suppose Alice is malicious. Even then, we want her real-world view of the interaction to be approximately the same as her view in the ideal world.

The naive picture (Figure 18) has Alice and Bob in the real world running the protocol, and Alice plus the judge in the ideal world. We want these two scenarios to be statistically close (\approx_ϵ).

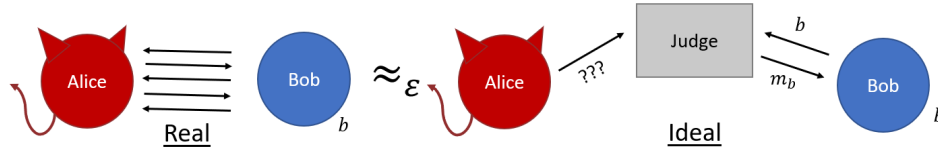


Figure 18: A first attempt at modeling security against malicious Alice.

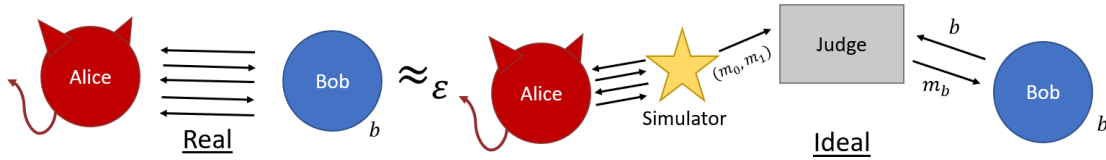


Figure 19: Security against malicious Alice via a simulator.

But there is a clear mismatch: in the ideal world Alice talks only to the judge, while in the real world she carries out a full protocol with Bob. We bridge this with a *simulator*.

The simulator plays the role of Bob in interacting with malicious Alice, but it must also *extract* Alice’s inputs (m_0, m_1) and pass them to the judge — all without knowing Bob’s choice b . We require Alice’s “view” (her record of the protocol so far at every step) in the simulator-world to be indistinguishable from her view in the real world.

Why this implies security. Because the simulator never sees b , Alice’s view in the simulator-world is statistically independent of b . By indistinguishability, the same holds for Alice’s view in the real-world. Hence Alice cannot learn anything about b .

12.5 Security Against Malicious Receivers

The construction is symmetric: a simulator on Bob’s side extracts the input from a malicious Bob and passes it to the judge.

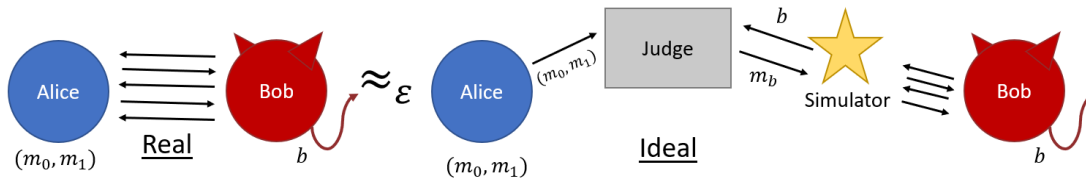


Figure 20: Security scenario for malicious Bob.

Reasoning as before, the simulator’s view is independent of m_{1-b} , so Bob’s view — which by indistinguishability matches the simulator’s — is also (nearly) independent of m_{1-b} .

⁸Technically, the relevant quantity is a variant of H_{\min} from [Ren05], but the intuition is the same.

12.6 Attempt: Information-Theoretically Secure Quantum OT

It is natural to ask whether quantum resources let us implement OT with information-theoretic security, in analogy with QKD. As stated this is hard, but we can aim for a relaxed primitive: a pair of “effective channels” such that Bob can read at most one of the two channels’ contents. This is sometimes called the “1-out-of-2 channels” primitive (Figure 21), and it captures the core idea behind 1-out-of-2 OT.

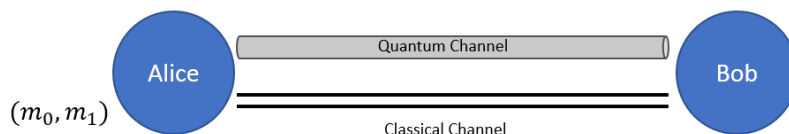


Figure 21: The setup for the 1-out-of-2-channels primitive. We use a single quantum channel and a single classical channel, but interpret them as implementing two logical channels.

Unlike QKD, the classical channel here does not need to be authenticated: there is no third-party eavesdropper.

Protocol via conjugate coding.

1. Alice samples $x, \theta \xleftarrow{\$} \{0, 1\}^n$, encodes $|\psi_i\rangle = |x_i\rangle_{\theta_i}$, and sends $\{|\psi_i\rangle\}_{i \in [n]}$ over the quantum channel.
2. Bob samples $\theta'_i \xleftarrow{\$} \{0, 1\}$ for each i , measures $|\psi_i\rangle$ in basis θ'_i , and announces “I measured” to Alice.
3. Alice reveals $\{\theta_i\}_{i \in [n]}$ to Bob.
4. Bob (privately) sets $S_1 := \{i : \theta_i = \theta'_i\}$ — the indices where his guessed basis matched.
5. Bob sends $I_0, I_1 \subseteq [n]$ to Alice, where $b \xleftarrow{\$} \{0, 1\}$, $I_b = S_1$, and $I_{1-b} = [n] \setminus S_1$.
6. Alice sends $z_j = m_j \oplus \bigoplus_{i \in I_j} x_i$ for $j \in \{0, 1\}$.
7. Bob computes $m_b = z_b \oplus \bigoplus_{i \in I_b} y_i$. Since $y_i = x_i$ for $i \in I_b = S_1$, this recovers m_b .

If both parties are honest, Bob learns m_b but not m_{1-b} (on the indices in I_{1-b} his measurement basis was wrong, so y_i is independent of x_i); Alice learns nothing about b , since it was chosen at random and never revealed.

Attack: malicious Bob delays measurement. The honest analysis assumes Bob measures in Step 2. But Bob is not forced to: he can wait until Alice reveals θ , then measure in the *correct* basis and recover *all* of x .

Dishonest Bob’s Attack:

1. Bob doesn’t measure in Step 2; he just announces that he has.
2. Alice sends $\{\theta_i\}_{i \in [n]}$.

3. Bob now measures $|\psi_i\rangle$ in basis θ_i , obtaining $y_i = x_i$ for all i .
4. Bob partitions $[n]$ arbitrarily as I_0, I_1 and proceeds with Step 5 of the honest protocol.
5. Given z_0, z_1 , Bob computes $m_j = z_j \oplus \bigoplus_{i \in I_j} x_i$ for both j .

So we must force Bob to measure in Step 2, and to prove that he did. The tool we need is a *commitment scheme*.

12.7 Commitment Schemes

Definition 10. A *commitment scheme* is a pair of algorithms (commit, decommit) satisfying:

1. **Hiding.** $\text{commit}(b) \rightarrow (\text{com}, \text{state})$ such that com does not reveal b .
2. **Binding.** $\text{decommit}(\text{com}, \text{state}, b) \in \{0, 1\}$ accepts only if com was generated using the claimed value b .

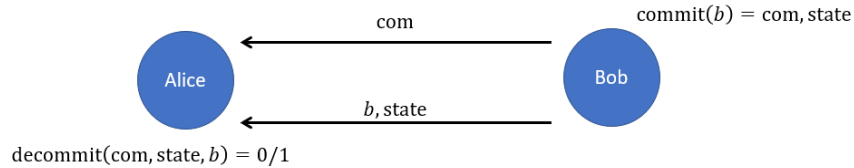


Figure 22: A commitment scheme: Bob commits to a value.

To prevent the attack above, Bob will commit to (y_i, θ'_i) for each i *before* learning θ . We make this precise and formal in the next lecture.

Key Takeaways. Strong agreement in QKD is obtained by running the basic protocol on top of an error-correcting code, so that Bob can deterministically recover Alice's message from his δn -close measurement string. Strong privacy follows from privacy amplification: a quantum-safe extractor shrinks Alice's bitstring to a shorter key, on which Eve has negligible advantage. Oblivious transfer, by contrast, is a two-party functionality where one party (Bob) selects which of two values to receive. Information-theoretic OT requires extra ingredients beyond pure quantum communication — specifically, a commitment scheme to force Bob to measure honestly. We develop this in the next lecture.

13 Quantum Oblivious Transfer – II

The goal of an oblivious transfer (OT) protocol is for a sender (Alice) to transmit one of several values to a receiver (Bob) such that:

- Alice does not learn which value Bob received.
- Bob does not learn the other values Alice holds.

As we saw in Lecture 12, a (classical or quantum) commitment scheme is a necessary primitive for any quantum OT (QOT) protocol that hopes to be secure against a malicious Bob.

13.1 Commitment Schemes

Definition 11. A *commitment scheme* is a pair of algorithms (C, V) :

$$\begin{aligned} C(m; r) &\rightarrow x, & m \in \{0, 1\}^p, r \in \{0, 1\}^\lambda, \\ V(x, m', r') &\rightarrow y, & y \in \{Accept, Reject\}, \end{aligned}$$

where m is the committed message and r is the randomness used. The scheme must satisfy:

- **Correctness:** for all m, r , $V(C(m; r), m, r) = Accept$.
- **Perfect Binding:** for every y , there is no pair (m, r, m', r') with $m \neq m'$ such that $V(y, m, r) = V(y, m', r') = Accept$.
- **Computational Hiding:** for all m, m' , $C(m; r) \approx_{\text{negl}(\lambda)} C(m'; r)$.

Correctness ensures that any legitimate commitment opens correctly. *Binding* ensures that no commitment y admits two different valid openings; equivalently, every accepted commitment has a unique pre-image (m, r) . *Hiding* ensures that, for a computationally bounded receiver, commitments to two different messages are indistinguishable.

13.1.1 Indistinguishability

We formalize *computational indistinguishability* against quantum polynomial-size circuits \mathcal{D} . For hiding, we require

$$|\Pr[\mathcal{D}(C(m; r)) = 1] - \Pr[\mathcal{D}(C(m'; r)) = 1]| \leq \text{negl}(\lambda),$$

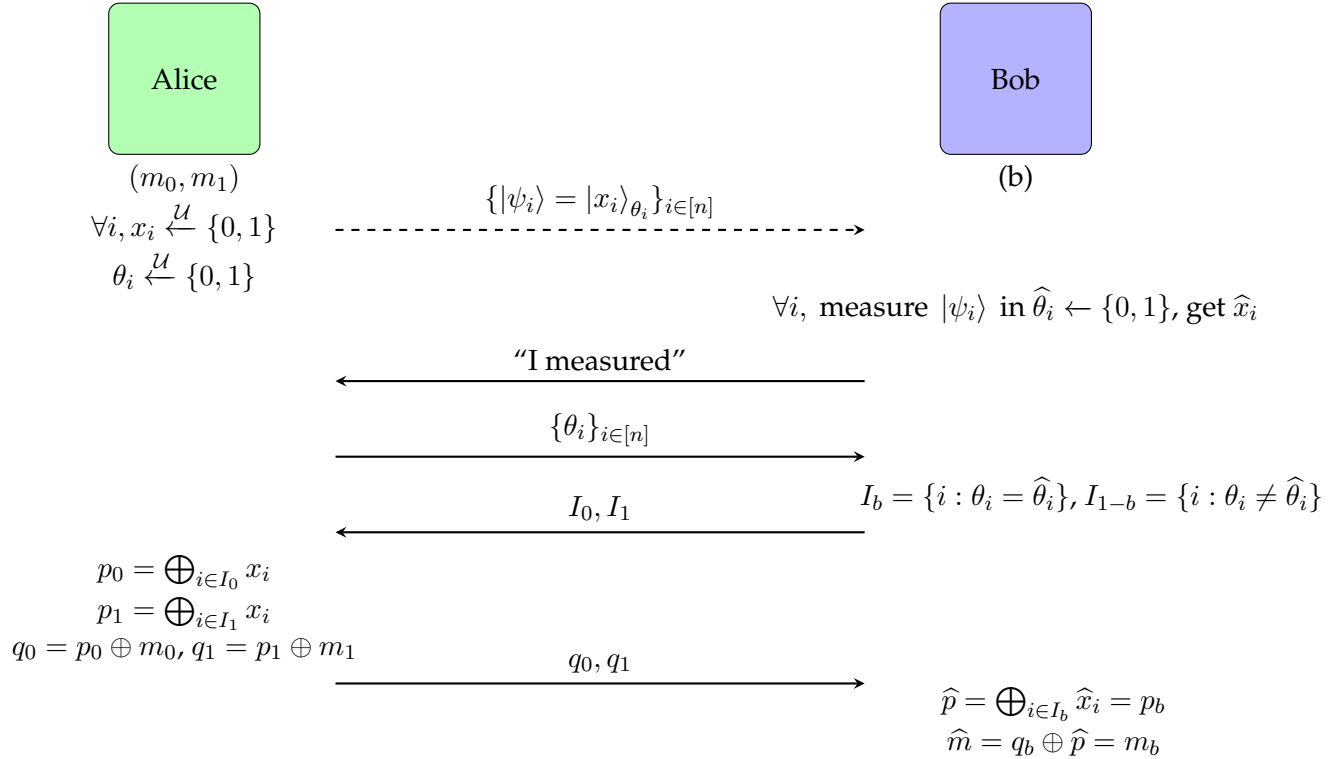
where $\lambda = |r|$ is the security parameter. A function $\varepsilon(\lambda)$ is *negligible* if it decays asymptotically faster than every inverse polynomial:

$$\forall c \in \mathbb{N}, \exists \lambda_0 \text{ such that } \forall \lambda > \lambda_0, \varepsilon(\lambda) < \frac{1}{\lambda^c}.$$

For instance, $2^{-\lambda}$ and $\lambda^{-\log \lambda}$ are negligible; λ^{-10} is not.

13.2 Quantum Oblivious Transfer

In Lecture 12 we presented a protocol that is correct, secure against a malicious Alice, but insecure against a malicious Bob. We reproduce it below.



The attack. A malicious Bob skips the measurement in Step 1, waits for Alice to reveal $\{\theta_i\}$, and then performs a *delayed measurement* in the correct basis θ_i (rather than $\hat{\theta}_i$). This recovers all of Alice's bits $\{x_i\}$. Bob can now arbitrarily partition $[n]$ into I_0, I_1 , and decrypt *both* $m_0 = q_0 \oplus \bigoplus_{i \in I_0} \hat{x}_i$ and $m_1 = q_1 \oplus \bigoplus_{i \in I_1} \hat{x}_i$.

We fix this by replacing the "I measured" announcement with a *cryptographic proof of measurement* using a commitment scheme.

13.2.1 Proof of Measurement

In the augmented protocol, after sampling $\hat{\theta}_i$ and obtaining \hat{x}_i , Bob commits to the pair $(\hat{x}_i, \hat{\theta}_i)$:

$$y_i = C((\hat{x}_i, \hat{\theta}_i); r_i).$$

He sends $\{y_i\}$ to Alice. Alice then samples a random check subset $T \subseteq [n]$ with $|T| = n/2$, and asks Bob to *open* the commitments on T , i.e. send $\{((\hat{x}_i, \hat{\theta}_i), r_i)\}_{i \in T}$. Alice verifies two conditions:

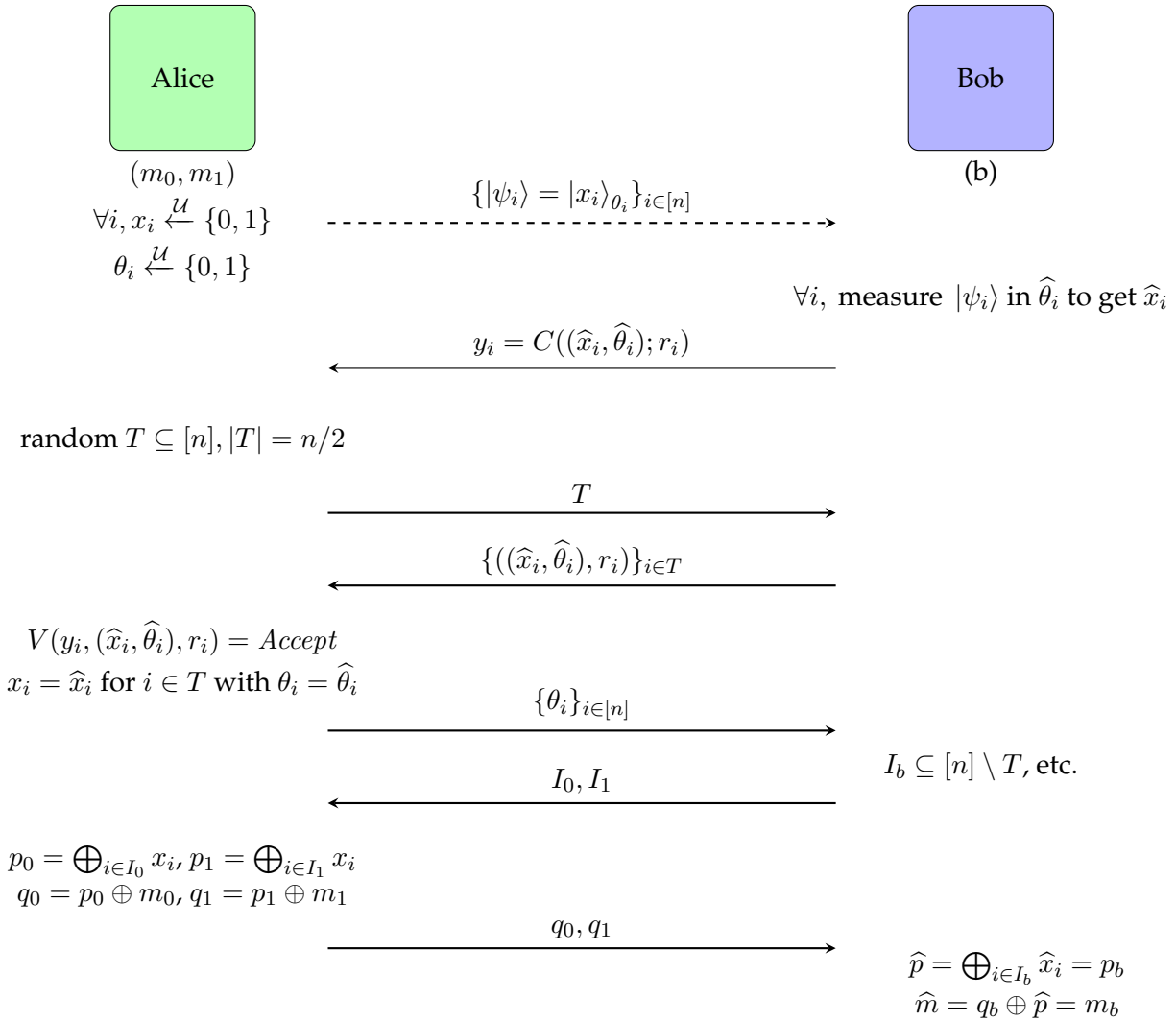
1. Openings are valid: $V(y_i, (\hat{x}_i, \hat{\theta}_i), r_i) = \text{Accept}$ for all $i \in T$.

2. Where Alice's and Bob's bases agree, the measurement outcome matches Alice's encoding:
 $x_i = \hat{x}_i$ for all $i \in T$ with $\theta_i = \hat{\theta}_i$.

If the checks pass, the protocol proceeds as before, with Alice sending $\{\theta_i\}_{i \in [n]}$. The sets I_0, I_1 are now partitions of $[n] \setminus T$ (rather than $[n]$):

$$I_b = \{i \in [n] \setminus T : \theta_i = \hat{\theta}_i\}, \quad I_{1-b} = \{i \in [n] \setminus T : \theta_i \neq \hat{\theta}_i\}.$$

The exclusion of T from the partition is crucial for Bob's privacy: since Bob revealed $\hat{\theta}_i$ for $i \in T$, including those indices would let Alice deduce his choice bit b .



13.2.2 Security Against Malicious Bob: Setup

We outline the simulator-based argument; the technical heart is the following theorem, proved next lecture.

Equivalent game. Instead of having Alice send classical states $|x_i\rangle_{\theta_i}$, we equip her with n EPR pairs $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ and have her send Bob the second half of each pair. From Bob’s perspective the two games are identical, because the reduced density matrix on Bob’s side is the maximally mixed state in either case.

Computationally unbounded simulator. The simulator (playing the role of Alice in the EPR-based game) is taken to be unbounded in computation.⁹ Because the commitment scheme is *binding*, every commitment Bob sends has a unique opening; an unbounded simulator can determine Bob’s committed values $(\hat{x}_i, \hat{\theta}_i)$ directly. The simulator also samples $T \subseteq [n]$ and $\theta_i \xleftarrow{\$} \{0, 1\}$ for $i \in T$, and measures her EPR halves at positions in $S = \{i : \theta_i = \hat{\theta}_i\}$ in basis θ_i . The remaining EPR halves at $[n] \setminus S$ are left unmeasured.

Rotated state. Define $|\psi'_i\rangle = H^{\hat{\theta}_i} |\psi_i\rangle$, applying a Hadamard rotation on the i -th register precisely when Bob’s measurement basis was the Hadamard basis. The following theorem (proved next lecture) is the central technical claim.

Theorem 13.1. *Given Bob’s commitments $\{(\hat{x}_i, \hat{\theta}_i)\}_{i \in [n]}$, Alice’s EPR halves $\{|\psi_i\rangle_A\}_{i \in [n]}$, a random $T \subseteq [n]$ with $|T| = n/2$, and a uniformly random $\{\theta_i \xleftarrow{\$} \{0, 1\}\}_{i \in T}$, the rotated states $\{|\psi'_i\rangle_A = H^{\hat{\theta}_i} |\psi_i\rangle_A\}_{i \in [n] \setminus T}$ are close to a superposition over computational-basis strings of low Hamming distance from Bob’s outcomes $(\hat{x}_1, \dots, \hat{x}_n)$.*

Conceptually, the theorem says that conditioned on Alice’s checks passing, Bob’s claimed measurement outcomes are nearly forced — so Bob cannot have skipped measurement without being caught.

Key Takeaways. A commitment scheme is a two-phase primitive — commit, then open — that gives a sender the power to “lock in” a value without revealing it (hiding) while remaining unable to change it later (binding). Plugging commitments into the QOT protocol forces Bob to measure honestly: he commits to his bases and outcomes before learning Alice’s bases, so a delayed-measurement attack is detected by Alice’s random subset check. Security against malicious Bob then reduces, via the standard EPR-pair rewriting trick, to a quantum sampling claim that we will prove in the next lecture.

⁹In simulation-based security, the simulator’s running time is typically less restricted than that of real-world parties; the meaningful notion is the indistinguishability of *Bob’s view* in the real and simulator worlds.

14 Quantum Oblivious Transfer – III

In the previous lectures, we showed that our QOT protocol is secure against a malicious sender (Alice) by exhibiting a simulator whose interaction with Alice is indistinguishable from Bob's, despite not knowing Bob's choice b . We also began the analysis of security against a malicious receiver (Bob), introducing commitment schemes for the proof-of-measurement step. We complete that analysis here and end with a short discussion of an application of OT.

14.1 Security Against Malicious Bob

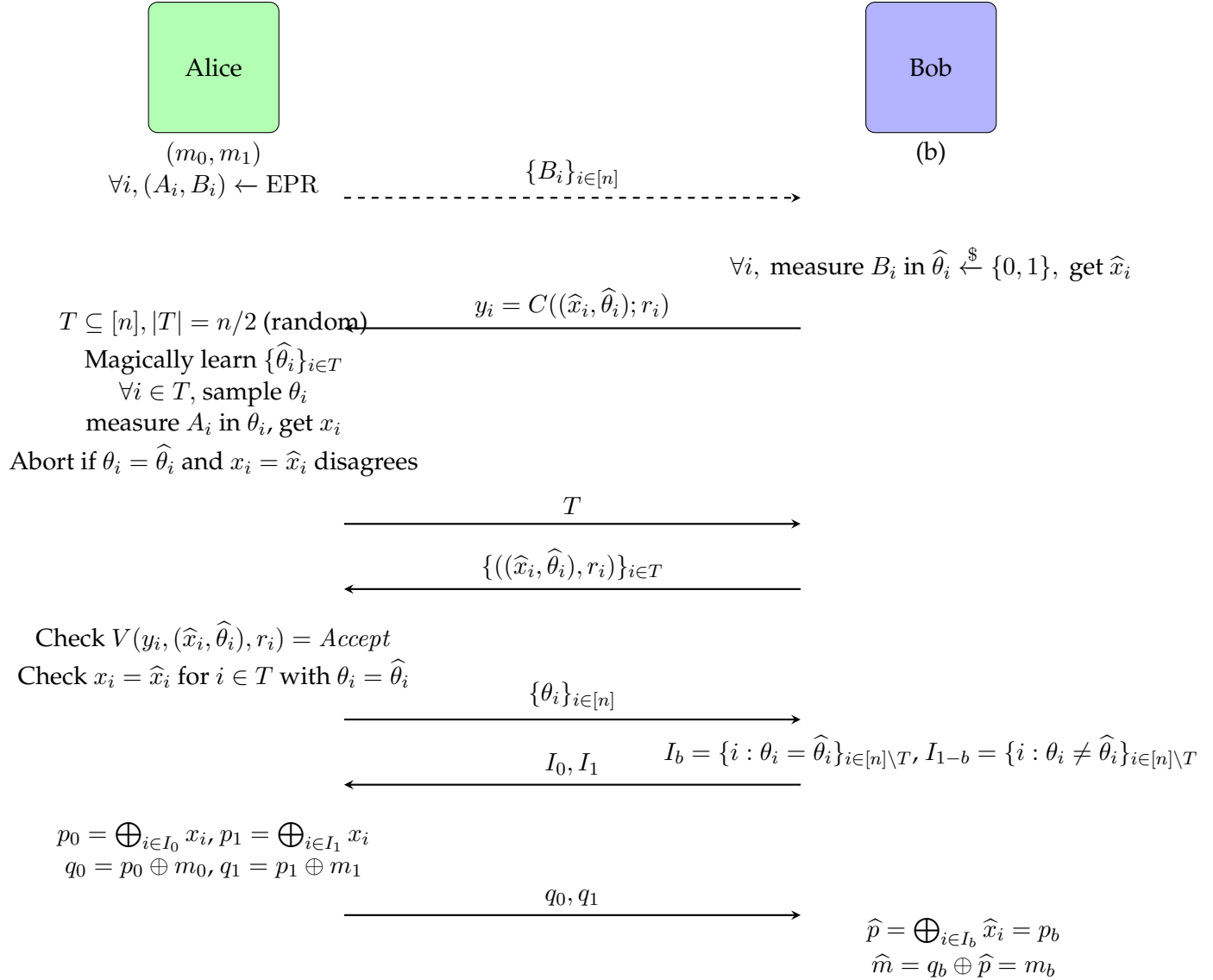
We will prove security against malicious Bob by analyzing a variant of QOT, denoted QOT*, that is indistinguishable from QOT on Bob's end. Because QOT* requires capabilities that Alice cannot actually possess (an "oracle" for Bob's committed bases), it is not implementable as a real-world protocol — but it suffices to establish indistinguishability and hence security.

14.1.1 The Variant QOT*

In QOT*, Alice has two extra powers:

1. She can learn $\hat{\theta}$, the bases Bob committed to, without Bob opening the commitments.
2. She sends Bob the second halves of n EPR pairs $|\psi_i\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ instead of computational/Hadamard-basis states.

Alice picks her own bases θ_i as before but *delays* her measurement until after she has learned $\hat{\theta}$. She then measures the indices in T in basis $\hat{\theta}$ and the remaining indices \bar{T} in θ . The full protocol is depicted below.



Equivalence on Bob's side. QOT* gives Alice strictly more information than the original protocol — she sees $\hat{\theta}$ and chooses her bases adaptively. However, Bob's view is unchanged. The only protocol-level difference is in Step 1: Bob receives the second half of an EPR pair instead of a fixed $|x_i\rangle_{\theta_i}$. But the reduced density matrix on Bob's side is the maximally mixed state in both cases, so Bob cannot distinguish them.

Hence, if we prove security in QOT*, the same security holds in the real-world QOT against any malicious Bob.

14.1.2 Security of QOT*

Define $|\psi'_i\rangle = H^{\hat{\theta}_i} A_i$: Alice's half of each EPR pair, rotated so that the measurement basis $\hat{\theta}_i$ becomes the computational basis. With perfect gates, $|\psi'_i\rangle$ is exactly $|\hat{x}_i\rangle$ when Bob did *not* lie about

his basis (i.e. when $\hat{\theta}_i$ accurately reports Bob’s measurement basis).

In QOT*, Alice already knows the committed $\hat{\theta}_i$ at the time she chooses her own bases, so she can sample $\theta_i = \hat{\theta}_i$ for $i \in T$ and verify that the EPR-induced measurement matches \hat{x}_i . Statistically, the probability that $|\psi'_i\rangle$ at non-tested positions ($i \in \bar{T}$) is far in Hamming distance from Bob’s committed \hat{x} -string is exponentially small: any mismatch on more than a few positions would have been detected by Alice’s random check.

More quantitatively, with exponentially high probability, the relevant non-tested bits of x_{1-b} — the side of the OT Bob is *not* supposed to learn — are predominantly at positions where $\theta_j \neq \hat{\theta}_j$. At those positions Bob’s measurement is in the wrong basis, so his recorded bit is uniformly random and independent of x_{1-b} .

Remark. The exact constants in “exponentially high” and “predominantly” come from Hoeffding- or Chernoff-type bounds; for the technical details see [BF12].

The upshot: in QOT*, the bits of x_{1-b} visible to Bob are nearly random and independent of the message m_{1-b} . Therefore $\bigoplus_{i \in I_{1-b}} x_i$ is effectively unknown to Bob, and the masked message $q_{1-b} = m_{1-b} \oplus \bigoplus_{i \in I_{1-b}} x_i$ does not reveal m_{1-b} .

14.2 Applications of OT

A central use of OT is *secure two-party computation*: two parties with private inputs a_1, \dots, a_k (Alice) and b_1, \dots, b_ℓ (Bob) jointly compute the output of any circuit C on input $(a_1, \dots, a_k, b_1, \dots, b_\ell)$ without revealing anything beyond C ’s output.

Trivial example: XOR. For $k = \ell = 1$ and $C = \oplus$, the output $a_1 \oplus b_1$ already determines both inputs given one of them. Hence the optimal protocol is simply to exchange bits.

AND. For $C = \wedge$, the output does not determine the inputs. Consider:

1. Alice computes $x_0 = a_1 \wedge 0 = 0$ and $x_1 = a_1 \wedge 1 = a_1$.
2. Bob runs 1-out-of-2 OT with Alice to receive x_{b_1} .
3. Bob outputs $x_{b_1} = a_1 \wedge b_1$.

Alice learns nothing (by OT security); Bob learns only $a_1 \wedge b_1$. In particular, if Alice’s input is $a_1 = 0$, the output 0 does not determine b_1 , so Bob’s privacy is preserved.

NOT. NOT is local: neither party needs the other’s input.

Putting it together: universal computation, but with intermediate leakage. NAND ($= \neg(\wedge)$) and OT together let us compute any Boolean circuit. However, this naive composition leaks intermediate wire values — a much larger leak than just the final output. For many functions, intermediate wire values uniquely determine the input.

Secret shares. To plug this leak, we use the following 2-out-of-2 secret sharing: a share of bit 0 is a uniformly random sample from $\{00, 11\}$, and a share of 1 is a uniformly random sample from $\{01, 10\}$. Either single share is uniform and independent of the secret bit; the XOR of the two shares recovers the secret.

For an XOR-circuit C , this immediately gives a leakage-free protocol:

1. Alice secret-shares each input bit a_i to get (s_i^A, s_i^B) and sends s_i^B to Bob.
2. Bob secret-shares each b_j similarly and sends s_j^A to Alice.
3. Each party now holds $k + \ell$ shares, one per input wire of C .
4. Alice and Bob evaluate C on their respective share strings (XOR gates act componentwise on shares, since \oplus is associative). At the end each holds a share of C (combined input).
5. They exchange final shares and XOR to recover the output.

The same approach extends to AND and NOT gates, but those steps require additional interaction (specifically, OT calls for each AND gate). See standard references on Yao's garbled circuits or the GMW protocol for the full construction.

Key Takeaways. A simulator-based proof of security against malicious Bob reduces to analyzing a hypothetical QOT* variant in which Alice has magical access to Bob's committed bases. The variant is indistinguishable from real QOT on Bob's side, and within QOT* a sampling argument shows that Bob's view of x_{1-b} is nearly random, hence m_{1-b} stays hidden. OT then lifts to universal secure two-party computation: with secret sharing handling XOR locally and OT calls handling AND, two parties can jointly compute any circuit while leaking only the output.

14.3 Further Reading

- Niek J. Bouman and Serge Fehr. *Sampling in a Quantum Population, and Applications*. [arXiv:0907.4246](https://arxiv.org/abs/0907.4246), specifically Section 5.
- Section 7.6 of [these lecture notes](#).

15 Quantum Random Oracles, Introduction to Encryption

In this lecture we introduce topics fundamental to the cryptographic constructions of the next several lectures. We first define the Random Oracle Model and use it to construct commitment schemes from hash functions; then we define symmetric encryption together with two notions of security (information-theoretic and CPA); finally we introduce the Learning With Errors (LWE) problem, which will furnish quantum-safe symmetric and public-key encryption schemes in subsequent lectures.

15.1 The Random Oracle Model

In Lecture 13 we saw that classical commitment schemes are essential for QOT, since the challenger must enforce Bob's commitment to his measurement outcomes. We previously *assumed* the existence of a commitment scheme; we now describe one concrete construction based on idealized hash functions.

Recall the requirements on a commitment scheme ($\text{Commit}, \text{Verify}$):

1. **(Perfect) Binding:** for every string str , there is no pair (m, r, m', r') with $m \neq m'$ and

$$\text{Verify}(\text{str}, m, r) = \text{Verify}(\text{str}, m', r') = 1.$$

2. **(Computational) Hiding:** for all m, m' , $\text{Commit}(m; r) \approx_{\text{negl}(\lambda)} \text{Commit}(m'; r)$.

Definition 12 (Random Oracle). Let k be the security parameter and $\ell_{\text{out}}(\cdot)$ a length function. A *random oracle* O is a function chosen uniformly at random from the set of all functions $\{0, 1\}^* \rightarrow \{0, 1\}^{\ell_{\text{out}}(k)}$. On a fresh input x , $O(x)$ is uniformly distributed in the output space; on a repeated input, O returns the same value as before.

Definition 13 (Hash Function). A *hash function* is a deterministic, efficiently computable function $h: \{0, 1\}^n \rightarrow \{0, 1\}^m$, typically with $m < n$.

A *cryptographic* hash function h is required to satisfy:

1. **Pre-image resistance:** given $t = h(m)$, it is hard to find any m' with $h(m') = t$.
2. **Second pre-image resistance:** given m_1 , it is hard to find $m_2 \neq m_1$ with $h(m_2) = h(m_1)$.
3. **Collision resistance:** it is hard to find any $m_1 \neq m_2$ with $h(m_1) = h(m_2)$.

In the *Random Oracle Model*, security analyses treat $h(\cdot)$ as a random oracle: every fresh query returns an independent uniform value. This idealization, while not realizable (any efficiently computable function is poly-size, but a random oracle requires exponentially many bits to specify), enables clean and provably secure constructions whose intuition often transfers to the standard model with carefully chosen hash functions.

Commitment from a Random Oracle. Let h be modeled as a random oracle. To commit to a bit m , Alice samples randomness r and outputs $\text{str} = h(m \parallel r)$. To verify, Bob accepts (str, m, r) iff $h(m \parallel r) = \text{str}$.

- *Binding*: since h is deterministic, two distinct inputs cannot produce the same str except by a hash collision; in the ROM, finding such a collision takes exponential time.
- *Hiding*: since h is a random oracle, the value $h(m \parallel r)$ is uniform conditional on the queried input, and hence reveals no information about m to anyone who has not queried $m \parallel r$.

15.2 Symmetric Encryption

Informally, symmetric (or *private-key*) encryption uses a shared secret key to convert plaintexts into ciphertexts that hide the plaintext from anyone without the key.

Definition 14 (Symmetric-Key Encryption). A *symmetric-key encryption scheme* is a triple of PPT algorithms $(\text{KeyGen}, \text{Enc}, \text{Dec})$:

- $\text{KeyGen}(1^\lambda; r) \rightarrow sk$: takes security parameter λ and randomness r , outputs a secret key sk .
- $\text{Enc}(sk, m; r') \rightarrow ct$: takes a key sk , message m , and randomness r' , outputs a ciphertext ct .
- $\text{Dec}(sk, ct) \rightarrow m$ or \perp : deterministic, given sk and ct outputs either the recovered plaintext or a failure symbol.

Correctness. *Statistical correctness* requires

$$\Pr_{r,r'}[\text{Dec}(sk, \text{Enc}(sk, m; r')) = m] \geq 1 - \text{negl}(\lambda).$$

Perfect correctness replaces the inequality with equality.

15.3 Notions of Security

15.3.1 Information-Theoretic Security

Definition 15. A scheme $(\text{KeyGen}, \text{Enc}, \text{Dec})$ with key space \mathcal{K} , message space \mathcal{M} , ciphertext space \mathcal{C} is *information-theoretically secure* if for all $m, m' \in \mathcal{M}$ and $c \in \mathcal{C}$,

$$\Pr_{k \leftarrow \mathcal{K}}[\text{Enc}(k, m) = c] = \Pr_{k \leftarrow \mathcal{K}}[\text{Enc}(k, m') = c].$$

Information-theoretic security says that the ciphertext distribution is identical regardless of the plaintext, so even an unbounded adversary learns nothing.

15.3.2 One-Time Pad

Definition 16. The *One-Time Pad* (OTP) is the symmetric scheme with $\mathcal{K} = \mathcal{M} = \mathcal{C} = \{0, 1\}^n$ defined by

- KeyGen : $sk \xleftarrow{\$} \{0, 1\}^n$.
- $\text{Enc}(sk, m) = sk \oplus m$.
- $\text{Dec}(sk, ct) = sk \oplus ct$.

Correctness of OTP. For any sk, m ,

$$\text{Dec}(sk, \text{Enc}(sk, m)) = sk \oplus (sk \oplus m) = m.$$

Information-theoretic security of OTP. We use the equivalent formulation $\Pr[M = m \mid C = c] = \Pr[M = m]$. Compute

$$\Pr[C = c] = \sum_{m'} \Pr[C = c \mid M = m'] \Pr[M = m'] = \sum_{m'} \Pr[sk = c \oplus m'] \Pr[M = m'] = \frac{1}{2^n},$$

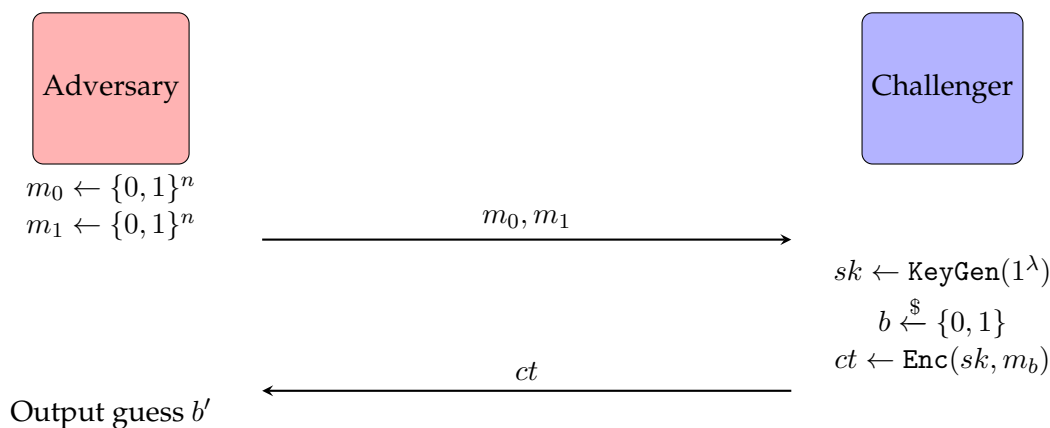
where we used $\Pr[sk = c \oplus m'] = 2^{-n}$ for each m' since sk is uniform. Applying Bayes,

$$\Pr[M = m \mid C = c] = \frac{\Pr[sk = c \oplus m] \Pr[M = m]}{\Pr[C = c]} = \frac{2^{-n} \Pr[M = m]}{2^{-n}} = \Pr[M = m].$$

The OTP has two limitations: the key must be at least as long as the message, and (more critically) keys can be used *only once*. We address the latter with the CPA notion below.

15.3.3 CPA Security: Single Message

The relaxed notion of security against a computationally bounded adversary is captured by the game between adversary \mathcal{A} (a probabilistic/quantum polynomial-time Turing machine) and challenger \mathcal{C} :



Definition 17 (CPA Security). A symmetric encryption scheme is *CPA-secure* if for all $m_0, m_1 \in \{0, 1\}^n$ and all PPT/QPT adversaries \mathcal{A} ,

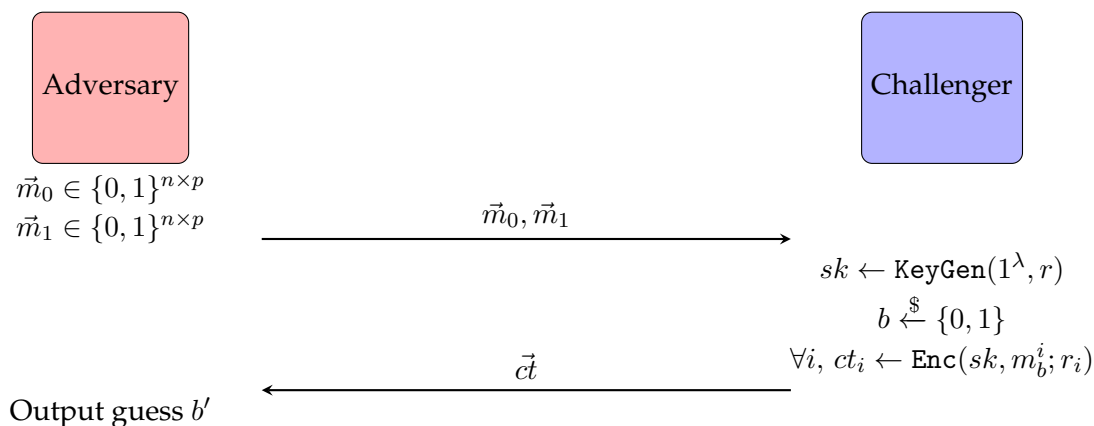
$$\left| \Pr_{sk}[\mathcal{A}(\text{Enc}(sk, m_0)) = 1] - \Pr_{sk}[\mathcal{A}(\text{Enc}(sk, m_1)) = 1] \right| = \text{negl}(\lambda).$$

Equivalently, $\Pr_{sk, b}[\mathcal{A}(\text{Enc}(sk, m_b)) = b] \leq \frac{1}{2} + \text{negl}(\lambda)$.

The adversary has negligible advantage over a random guess.

15.3.4 CPA Security: Multi-Message

When the same key is reused, we strengthen the definition. The game now has \mathcal{A} submit vectors \vec{m}_0, \vec{m}_1 of messages:



Definition 18 (Multi-Message Security). A symmetric encryption scheme is *multi-message secure* if for all \vec{m}_0, \vec{m}_1 and all PPT/QPT \mathcal{A} ,

$$\left| \Pr_{sk}[\mathcal{A}(\vec{ct}_0) = 1] - \Pr_{sk}[\mathcal{A}(\vec{ct}_1) = 1] \right| = \text{negl}(\lambda),$$

where \vec{ct}_b denotes the vector of per-message encryptions of \vec{m}_b under sk .

Why OTP fails multi-message security. Take \vec{m}_0 with two equal messages and \vec{m}_1 with two different ones. Then

$$ct_1 \oplus ct_2 = (sk \oplus m_b^1) \oplus (sk \oplus m_b^2) = m_b^1 \oplus m_b^2.$$

If the XOR is 0^n , the adversary outputs 0; otherwise 1. Advantage: 1.

Relationship. Multi-message security implies CPA security (the special case of length-1 vectors). The converse is false, as the OTP example shows: CPA-secure but not multi-message secure.

15.4 Learning With Errors

The Learning With Errors (LWE) problem is a workhorse computational problem used to build quantum-safe encryption.

Motivation. Consider an $n \times n$ system of linear equations over \mathbb{Z}_p :

$$\begin{aligned} a_1 \vec{x} + b_1 &= y_1, \\ &\vdots \\ a_n \vec{x} + b_n &= y_n, \end{aligned}$$

with $a_i, b_i \in \mathbb{Z}_p$. Gaussian elimination solves this in polynomial time. But if we perturb each equation by a small error $e_i \in \{-1, 0, 1\}$,

$$\begin{aligned} a_1 \vec{x} + e_1 &\equiv y_1 \pmod{p}, \\ &\vdots \\ a_n \vec{x} + e_n &\equiv y_n \pmod{p}, \end{aligned}$$

the problem becomes much harder: a brute-force search has to try 3^n error patterns, suggesting at least exponential time without a clever algorithm. The LWE assumption posits exactly this hardness, against quantum adversaries.

15.4.1 Search-LWE

Definition 19 (Search-LWE Assumption). For parameters n , prime $q \sim 2^n$, $m = \text{poly}(n)$, and error distribution χ (e.g. uniform on $\{-1, 0, 1\}$):

Sample $\vec{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $A \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$, $\vec{e} \xleftarrow{\$} \chi^m$, and let $\vec{b} = A\vec{s} + \vec{e}$.
For every QPT adversary \mathcal{A} ,

$$\Pr[\mathcal{A}(A, \vec{b}) = \vec{s}' \text{ such that } A\vec{s}' + \vec{e}' = \vec{b} \text{ for some } \vec{e}' \in \{-1, 0, 1\}^m] = \text{negl}(n).$$

15.4.2 Decisional LWE

Definition 20 (Decisional LWE Assumption). With $(A, \vec{s}, \vec{e}, \vec{b})$ as above and $\vec{u} \xleftarrow{\$} \mathbb{Z}_q^m$:

$$|\Pr[\mathcal{A}(A, \vec{b}) = 1] - \Pr[\mathcal{A}(A, \vec{u}) = 1]| = \text{negl}(n).$$

Why DLWE is non-trivial. A computationally unbounded distinguisher can succeed: the pair (A, \vec{b}) comes from a distribution of size $q^n \cdot |\text{supp}(\chi)|^m$, while (A, \vec{u}) has support q^m . For $m \gg n$ and reasonable error distribution support, $q^m \gg q^n |\text{supp}(\chi)|^m$, so the two distributions differ significantly. The DLWE assumption says no QPT can detect the gap.

Equivalence. Search and decisional LWE are polynomial-time equivalent for appropriate parameters. We use DLWE in the next lecture to build a multi-message-secure symmetric encryption scheme.

Key Takeaways. The Random Oracle Model is a useful (but idealized) tool that delivers commitment schemes essentially for free from a hash function. Symmetric encryption admits both an information-theoretic notion (achieved by the One-Time Pad with one-time use) and computational notions like CPA security and the stronger multi-message security; the OTP is CPA-secure but *not* multi-message secure, because XOR of two ciphertexts cancels the key. The Learning With Errors problem — search or decisional — provides a quantum-hard assumption from which we will build multi-message-secure symmetric encryption (and, later, public-key encryption) in the lectures to come.

16 Encrypting Classical Bits and Quantum States

In this lecture, we use the (Decisional) Learning With Errors assumption introduced in the previous lecture to build a symmetric-key encryption scheme that is multi-message secure against quantum polynomial-time adversaries. We then extend the construction to encrypt arbitrary quantum states using the Quantum One-Time Pad (QOTP).

16.1 Multi-Message Security: Recap

16.1.1 The Adversary-Challenger Game

Recall the multi-message indistinguishability game between an adversary \mathcal{A} and a challenger \mathcal{C} .

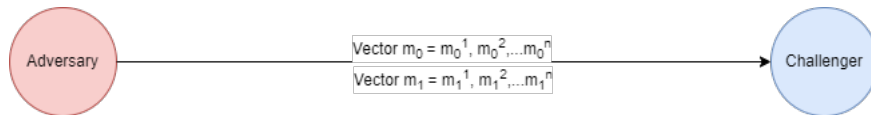


Figure 23: \mathcal{A} sends two message vectors \vec{m}_0, \vec{m}_1 to the challenger.

The adversary samples two vectors of messages $\vec{m}_0 = (m_0^1, \dots, m_0^n)$ and $\vec{m}_1 = (m_1^1, \dots, m_1^n)$ and sends them to the challenger.

The challenger samples a random bit $b \xleftarrow{\$} \{0, 1\}$, a key $k \xleftarrow{\$} \{0, 1\}^\lambda$, and returns the encryption of \vec{m}_b under k :

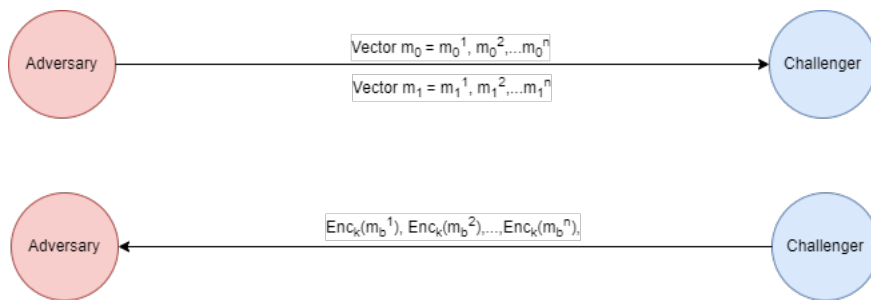


Figure 24: The challenger encrypts \vec{m}_b and returns the ciphertext vector.

The adversary outputs a guess \tilde{b} .

Definition 21 (Multi-Message Security). A symmetric encryption scheme satisfies *multi-message security* if, for every quantum polynomial-time adversary \mathcal{A} ,

$$\Pr[\tilde{b} = b] \leq \frac{1}{2} + \text{negl}(\lambda).$$

16.1.2 Why the One-Time Pad Fails for Multiple Messages

Recall the classical *one-time pad* (OTP): the encryption of message m under key k is $\text{Enc}(k, m) = k \oplus m$. The scheme is information-theoretically secure when the key is uniformly random and used exactly once.

If, however, the same key is used to encrypt two messages, an adversary obtains $c_1 = k \oplus m^1$ and $c_2 = k \oplus m^2$, hence $c_1 \oplus c_2 = m^1 \oplus m^2$, which is independent of the key.

Concrete attack. Take $\vec{m}_0 = (0^\lambda, 1^\lambda)$ and $\vec{m}_1 = (0^\lambda, 0^\lambda)$.

- If $b = 0$, the ciphertexts are $(k, k \oplus 1^\lambda)$, which XOR to 1^λ .
- If $b = 1$, the ciphertexts are (k, k) , which XOR to 0^λ .

The adversary computes $c_1 \oplus c_2$ and outputs $\tilde{b} = 0$ if the result is 1^λ , else $\tilde{b} = 1$, succeeding with probability 1.

Resolution: pseudo-random keys. The OTP attack hinges on key reuse, but generating a fresh truly random key for every message is impractical. The remedy is to start from a single uniformly random key and generate a sequence of *pseudo-random* keys, one per message, that no quantum polynomial-time adversary can distinguish from independent uniform keys. Each pseudo-random key is used as a one-time pad on a single message.

16.2 Multi-Message Encryption from LWE

16.2.1 Reviewing the LWE Assumption

Recall the (Decisional) Learning With Errors assumption: for parameters n, m, q , and an error distribution χ ,

$$(A, A\vec{s} + \vec{e}) \approx_c (A, \vec{b}),$$

where $A \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$, $\vec{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\vec{e} \xleftarrow{\$} \chi^m$, and $\vec{b} \xleftarrow{\$} \mathbb{Z}_q^m$. Here \approx_c denotes computational indistinguishability against quantum polynomial-time adversaries. The dimensions are

$$A : m \times n, \quad \vec{s} : n \times 1, \quad \vec{e} : m \times 1, \quad \vec{b} : m \times 1.$$

For the indistinguishability to be meaningful, the entries of \vec{e} should have small norm, e.g. in $\{-1, 0, 1\}$.

16.2.2 Single-Message Encryption from LWE

We first construct a single-message scheme; multi-message security will follow.

Construction. The secret key is a uniformly random LWE secret $\vec{s} \in \mathbb{Z}_q^n$. To encrypt a single bit $\mu \in \{0, 1\}$:

1. Sample $\vec{a} \xleftarrow{\$} \mathbb{Z}_q^{1 \times n}$ and $e \xleftarrow{\$} \chi$.
2. Output the ciphertext $(\vec{a}, \vec{a}\vec{s} + e + \mu \cdot \lfloor q/2 \rfloor) \bmod q$.

Decryption. Given ciphertext (\vec{a}, c) with $c = \vec{a}\vec{s} + e + \mu\lfloor q/2\rfloor$, compute

$$c - \vec{a}\vec{s} = e + \mu\lfloor q/2\rfloor.$$

If $\mu = 0$, the result is the small error e ; if $\mu = 1$, it is $\lfloor q/2\rfloor + e$. As long as $|e| < q/4$, the two cases can be distinguished: $\mu = 1$ iff $|c - \vec{a}\vec{s}| \in [q/4, 3q/4] \pmod{q}$, and $\mu = 0$ otherwise.

Why scale μ by $\lfloor q/2\rfloor$? If we had instead set $c = \vec{a}\vec{s} + e + \mu$, decryption would yield $\mu + e$, in which a small error term would be indistinguishable from a non-zero message. Multiplying μ by $\lfloor q/2\rfloor$ places the two possible plaintexts at well-separated points in \mathbb{Z}_q , far enough apart that any error of magnitude less than $q/4$ does not cause confusion. This trick is used throughout LWE-based cryptography.

Security. Under the DLWE assumption,

$$(\vec{a}, \vec{a}\vec{s} + e + \mu\lfloor q/2\rfloor) \approx_c (\vec{a}, \vec{b} + \mu\lfloor q/2\rfloor),$$

where $\vec{b} \xleftarrow{\$} \mathbb{Z}_q$. But since \vec{b} is uniform, so is $\vec{b} + \mu\lfloor q/2\rfloor$, and the latter distribution is independent of μ . Hence the ciphertext computationally hides μ .

16.2.3 Multi-Message Encryption

The same secret key \vec{s} can safely be used to encrypt many messages, because every ciphertext samples its own fresh randomness (\vec{a}_i, e_i) .

Encryption of a vector $\vec{\mu} = (\mu_1, \dots, \mu_p)$. For each $i \in [p]$, sample $\vec{a}_i \xleftarrow{\$} \mathbb{Z}_q^{1 \times n}$ and $e_i \xleftarrow{\$} \chi$, and output

$$ct_i = (\vec{a}_i, \vec{a}_i\vec{s} + e_i + \mu_i\lfloor q/2\rfloor).$$

Equivalently, in matrix form,

$$ct = (A, A\vec{s} + \vec{e} + \vec{\mu} \cdot \lfloor q/2\rfloor),$$

where A is $p \times n$, \vec{e} and $\vec{\mu}$ are in \mathbb{Z}_q^p .

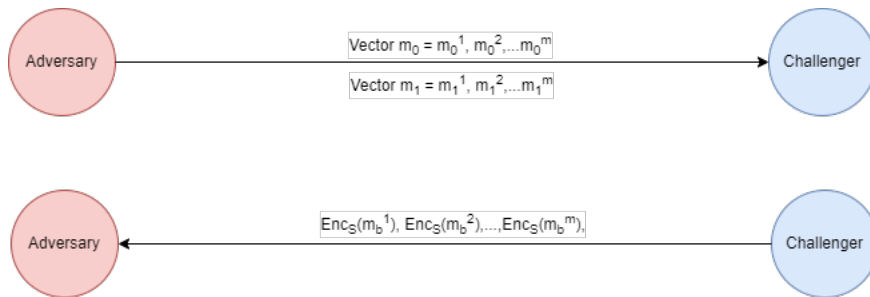


Figure 25: Multi-message security game; μ in the figure corresponds to a single bit of \vec{m} .

Multi-message security. By DLWE, the pair $(A, A\vec{s} + \vec{e})$ is computationally indistinguishable from (A, \vec{b}) for \vec{b} uniform in \mathbb{Z}_q^p . Adding $\vec{\mu} \cdot \lfloor q/2 \rfloor$ to a uniformly random vector yields another uniformly random vector, so

$$(A, A\vec{s} + \vec{e} + \vec{\mu}\lfloor q/2 \rfloor) \approx_c (A, \vec{b}),$$

and the right-hand side is independent of $\vec{\mu}$. Therefore the adversary cannot distinguish encryptions of \vec{m}_0 and \vec{m}_1 except with negligible advantage.

16.3 Quantum One-Time Pad

We now extend encryption to *quantum* states. The goal: given a (possibly mixed) state ρ and a classical key, output a ciphertext that is information-theoretically indistinguishable from the maximally mixed state $\mathbb{I}/2$ over the choice of key.

Construction. Sample two classical bits $a, b \stackrel{\$}{\leftarrow} \{0, 1\}$ as the key. To encrypt a single-qubit state ρ , output

$$\text{Enc}((a, b), \rho) = X^a Z^b \rho (X^a Z^b)^\dagger.$$

Recall the Pauli matrices

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

Enumerating the four key values:

$$\begin{aligned} \text{Enc}((0, 0), \rho) &= \rho, \\ \text{Enc}((0, 1), \rho) &= Z\rho Z^\dagger, \\ \text{Enc}((1, 0), \rho) &= X\rho X^\dagger, \\ \text{Enc}((1, 1), \rho) &= XZ\rho(XZ)^\dagger. \end{aligned}$$

Decryption with key (a, b) applies $(X^a Z^b)^\dagger$ on both sides:

$$\text{Dec}((a, b), \sigma) = (X^a Z^b)^\dagger \sigma (X^a Z^b).$$

Claim. Averaging the encryption over a uniformly random key yields the maximally mixed state:

$$\frac{1}{4} \sum_{a,b \in \{0,1\}} (X^a Z^b) \rho (X^a Z^b)^\dagger = \frac{\mathbb{I}}{2}. \quad (16.1)$$

Proof. Write $\rho = \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix}$ with $\alpha + \delta = \text{tr}(\rho) = 1$. We compute each term in the average:

$$\begin{aligned} Z\rho Z^\dagger &= \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = \begin{bmatrix} \alpha & -\beta \\ -\gamma & \delta \end{bmatrix}, \\ X\rho X^\dagger &= \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} \delta & \gamma \\ \beta & \alpha \end{bmatrix}, \\ (XZ)\rho(XZ)^\dagger &= \begin{bmatrix} \delta & -\gamma \\ -\beta & \alpha \end{bmatrix}. \end{aligned}$$

A useful mnemonic: a Z -conjugation negates the off-diagonal entries; an X -conjugation reflects across the anti-diagonal; combining these gives the fourth case. Summing the four matrices, the off-diagonal entries cancel, and the diagonal entries become $2(\alpha + \delta) = 2$ each:

$$\frac{1}{4} \begin{bmatrix} 2(\alpha + \delta) & 0 \\ 0 & 2(\alpha + \delta) \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \frac{\mathbb{I}}{2}. \quad \square$$

Interpretation. The QOTP is the quantum analogue of the classical one-time pad: with two random classical bits, one can encrypt an arbitrary single-qubit state so that the ciphertext is information-theoretically indistinguishable from the maximally mixed state. Anyone holding the two classical bits can deterministically recover the original ρ .

16.4 Combining the QOTP with LWE Encryption

The QOTP requires 2 fresh random bits per qubit of plaintext. To encrypt many qubits with a single short classical key, encrypt the QOTP keys themselves under the LWE-based multi-message scheme. The full encryption of a single-qubit state ρ is then

$$\text{Enc}(sk, \rho) = \left(X^a Z^b \rho (X^a Z^b)^\dagger, \text{Enc}_{\text{LWE}}(sk, (a, b)) \right),$$

where $(a, b) \stackrel{\$}{\leftarrow} \{0, 1\}^2$ is fresh for every plaintext. The recipient decrypts the classical component first to recover (a, b) , then applies $(X^a Z^b)^\dagger$ on both sides of the quantum component. This construction will be the foundation for quantum public-key encryption in the lectures that follow.

Key Takeaways. The classical one-time pad is broken by reuse, but a single LWE secret combined with fresh randomness per ciphertext yields a multi-message-secure symmetric encryption scheme. The Quantum One-Time Pad encrypts a single qubit using two classical bits so that the ciphertext is exactly the maximally mixed state. Combining LWE encryption (to hide the QOTP keys) with the QOTP (to mask the quantum plaintext) gives an encryption scheme for quantum messages whose security reduces to DLWE.

17 Private-Key Encryption for Classical Bits and Qubits

In this lecture we wrap up our discussion of the Quantum One-Time Pad (QOTP) and use it to build a private-key encryption scheme that encrypts arbitrary qubits, building on the LWE-based scheme for classical bits introduced in the last lecture.

17.1 The Quantum One-Time Pad and Pauli Gates

Recall the classical one-time pad: given a key $k \stackrel{\$}{\leftarrow} \{0, 1\}$ and bit $c \in \{0, 1\}$, the encryption is

$$\text{OTP}(c, k) = c \oplus k.$$

We now construct a quantum analogue. We will use the Pauli X and Z gates, which respectively rotate the Bloch sphere by angle π around the x - and z -axes; their action will become geometrically clear below.

17.1.1 The Pauli X Gate

The Pauli X gate is the quantum analogue of the classical NOT in the computational basis:

$$X = \sigma_x = \text{NOT} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

Its action on basis states:

$$X|0\rangle = |1\rangle, \quad X|1\rangle = |0\rangle, \quad X(\alpha|0\rangle + \beta|1\rangle) = \alpha|1\rangle + \beta|0\rangle.$$

On Hadamard-basis states:

$$X|+\rangle = |+\rangle, \quad X|-\rangle = -|-\rangle.$$

The second identity follows from $X\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) = \frac{|1\rangle - |0\rangle}{\sqrt{2}} = -|-\rangle$.

In the computational basis, X realizes a quantum one-time pad on *classical* qubits: sample $a \stackrel{\$}{\leftarrow} \{0, 1\}$ and output $X^a|c\rangle$. With probability $\frac{1}{2}$ this flips $|c\rangle$; with probability $\frac{1}{2}$ it leaves it intact.

17.1.2 The Pauli Z Gate

The X -based one-time pad fails in the Hadamard basis: since $X|+\rangle = |+\rangle$ and $X|-\rangle = -|-\rangle$, applying X^a leaves $|+\rangle$ and $|-\rangle$ globally indistinguishable, so it does not hide whether the underlying state is $|+\rangle$ or $|-\rangle$.

The remedy is the Pauli Z gate, which flips $|+\rangle \leftrightarrow |-\rangle$:

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad Z|+\rangle = |-\rangle, \quad Z|-\rangle = |+\rangle.$$

For Hadamard-basis qubits, sampling $b \stackrel{\$}{\leftarrow} \{0, 1\}$ and outputting $Z^b|\psi\rangle$ gives a one-time pad.

17.1.3 The Quantum One-Time Pad in Full Generality

For an arbitrary qubit (which may be in superposition or mixed), we combine both:

$$\text{QOTP}_{a,b}(|\psi\rangle) = X^a Z^b |\psi\rangle, \quad (a, b) \stackrel{\$}{\leftarrow} \{0, 1\}^2.$$

As shown in Lecture 16, averaging over the four key values yields the maximally mixed state $\frac{\mathbb{I}}{2}$ for any input. Hence the QOTP achieves *information-theoretic* security with 2 random classical bits per qubit.

17.1.4 Geometric Interpretation: the Bloch Sphere

The Bloch sphere is the standard geometric depiction of a pure single-qubit state (Figure 26). The state $|0\rangle$ sits on the positive z -axis, $|1\rangle$ on the negative z -axis; $|+\rangle$ on the positive x -axis, $|-\rangle$ on the negative x -axis. An arbitrary pure state is parameterized by two angles (θ, ϕ) as

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

where θ is the polar angle from the z -axis and ϕ is the azimuthal angle in the xy -plane. Since the overall phase is unobservable, we have used up our remaining freedom to make the $|0\rangle$ amplitude real.

In this picture, X rotates the Bloch sphere by π around the x -axis (e.g. $|0\rangle \rightarrow |1\rangle$), and Z rotates by π around the z -axis (e.g. $|+\rangle \rightarrow |-\rangle$). The four-element QOTP samples one of $\{\mathbb{I}, X, Z, XZ\}$ uniformly at random, sending $|\psi\rangle$ to one of four positions related by these axis flips. Averaged over the key, the resulting density matrix is $\frac{\mathbb{I}}{2}$.

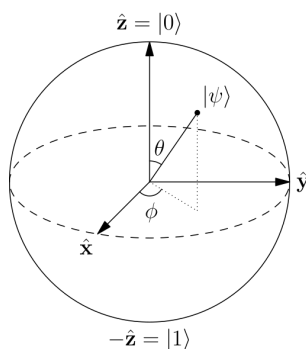


Figure 26: The Bloch sphere.

17.2 Private-Key Encryption

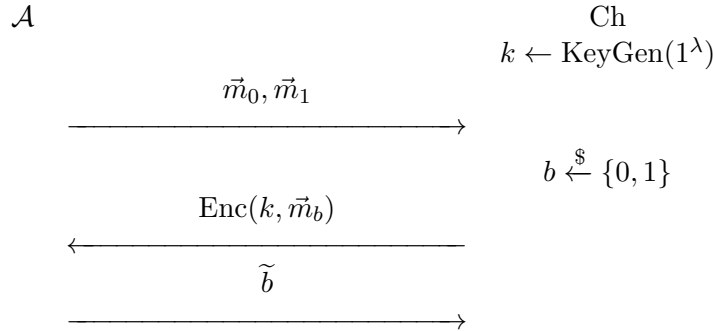
Definition 22. A *private-key encryption scheme* is a tuple of polynomial-time algorithms (KeyGen, Enc, Dec) satisfying:

- *Syntax.* KeyGen is probabilistic and outputs, on input 1^λ , a secret key k of length λ . Enc is probabilistic and takes (k, m) to a ciphertext ct . Dec is deterministic and takes (k, ct) to a message.

- *Correctness.* For every m and $k \in \text{supp}(\text{KeyGen}(1^\lambda))$,

$$\Pr[\text{Dec}(k, \text{Enc}(k, m)) = m] \geq 1 - \text{negl}(\lambda).$$

- *Security.* Consider the multi-message security game between adversary \mathcal{A} and challenger Ch:



For every QPT \mathcal{A} , $\Pr[\tilde{b} = b] \leq \frac{1}{2} + \text{negl}(\lambda)$.

17.2.1 One-Time vs. Multi-Message Security: Recap

The classical OTP achieves the above definition only when the adversary's vectors \vec{m}_0, \vec{m}_1 contain a single message each. When the same key is reused across multiple messages, the adversary can exploit $c_1 \oplus c_2 = m_1 \oplus m_2$ (independent of k) to win the game; concretely, $\vec{m}_0 = (0^\lambda, 1^\lambda)$ and $\vec{m}_1 = (1^\lambda, 0^\lambda)$ are perfectly distinguished. The LWE-based scheme from the previous lecture remedies this: each message gets its own fresh randomness, while the secret key \vec{s} is reused.

17.2.2 Encryption of Qubits

We now combine the QOTP with any quantum-secure classical private-key encryption scheme $(\text{KeyGen}_c, \text{Enc}_c, \text{Dec}_c)$ to obtain a private-key encryption scheme for qubits.

Construction.

- $\text{KeyGen}_Q(1^\lambda)$ runs $\text{KeyGen}_c(1^\lambda)$ to obtain a classical key k_c .
- $\text{Enc}_Q(k_c, \rho)$: sample $a, b \xleftarrow{\$} \{0, 1\}$, compute

$$\sigma = X^a Z^b \rho (X^a Z^b)^\dagger, \quad \text{ct} = \text{Enc}_c(k_c, (a, b)),$$

and output (σ, ct) .

- $\text{Dec}_Q(k_c, (\sigma, \text{ct}))$: run $\text{Dec}_c(k_c, \text{ct})$ to recover (a, b) , and output $(X^a Z^b)^\dagger \sigma (X^a Z^b)$.

The intuition is straightforward: the QOTP keys (a, b) are encrypted classically, and the quantum plaintext is masked by the corresponding Pauli. Anyone holding the classical key recovers (a, b) and undoes the masking.

17.2.3 Correctness

By correctness of the classical scheme, Dec_c recovers the exact (a, b) used in encryption. Substituting into the decryption formula,

$$\text{Dec}_Q(k_c, (\sigma, \text{ct})) = (X^a Z^b)^\dagger \sigma (X^a Z^b) = (X^a Z^b)^\dagger (X^a Z^b) \rho (X^a Z^b)^\dagger (X^a Z^b) = \rho,$$

since $(X^a Z^b)$ is unitary, hence $(X^a Z^b)^\dagger (X^a Z^b) = \mathbb{I}$.

17.2.4 Security

We argue informally; a formal hybrid-argument proof appears in the next lecture for the public-key analogue. The QOTP component σ contains no information about ρ in the absence of the key: averaged over uniform (a, b) , it equals the maximally mixed state. The classical component $\text{ct} = \text{Enc}_c(k_c, (a, b))$ hides (a, b) by the (multi-message) security of the classical scheme. By a hybrid argument, the adversary's view is computationally indistinguishable from one in which ct is the encryption of a fixed key (say $(0, 0)$), at which point the quantum component is statistically indistinguishable from $\frac{\mathbb{I}}{2}$ and reveals nothing about ρ .

Key Takeaways. The Pauli X gate hides classical-basis information; the Pauli Z gate hides Hadamard-basis information. Combining them yields the QOTP, which information-theoretically masks any single qubit using 2 random classical bits, with the four resulting states averaging to the maximally mixed state. By encrypting the 2-bit QOTP key with a quantum-secure classical scheme (such as the LWE-based scheme of Lecture 16), we obtain a private-key encryption scheme for quantum states that inherits the security of the underlying classical scheme.

18 Public-Key Encryption for Classical Bits and Qubits from LWE

In this lecture, we construct public-key encryption schemes whose security follows from the decisional LWE assumption. We begin by recalling the definition of public-key encryption, then show that for public-key encryption multi-message security follows from single-message security. We introduce Regev's scheme [Reg09] for encrypting classical bits, and finally show how any (quantum-secure) classical PKE can be promoted to a PKE that encrypts qubits.

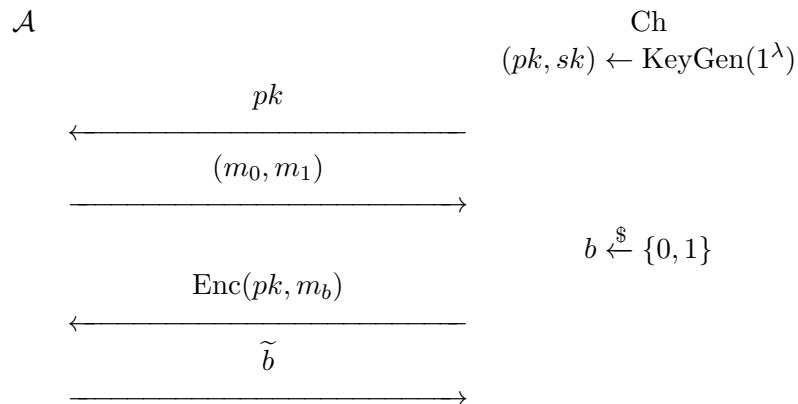
18.1 Definition of Public-Key Encryption

Definition 23. A *public-key encryption scheme* is a tuple of polynomial-time algorithms $(\text{KeyGen}, \text{Enc}, \text{Dec})$ satisfying:

- *Syntax.* KeyGen is probabilistic, takes 1^λ and outputs (pk, sk) . Enc is probabilistic, takes (pk, m) and outputs a ciphertext ct . Dec is deterministic, takes (sk, ct) and outputs a message.
- *Correctness.* For any (pk, sk) in the support of $\text{KeyGen}(1^\lambda)$ and any m ,

$$\Pr_r[\text{Dec}(sk, \text{Enc}(pk, m; r)) = m] \geq 1 - \text{negl}(\lambda).$$

- *Security.* Consider the following game between adversary \mathcal{A} and challenger Ch :



For every QPT \mathcal{A} , $\Pr[\tilde{b} = b] \leq \frac{1}{2} + \text{negl}(\lambda)$.

18.2 Single-Message vs. Multi-Message Security

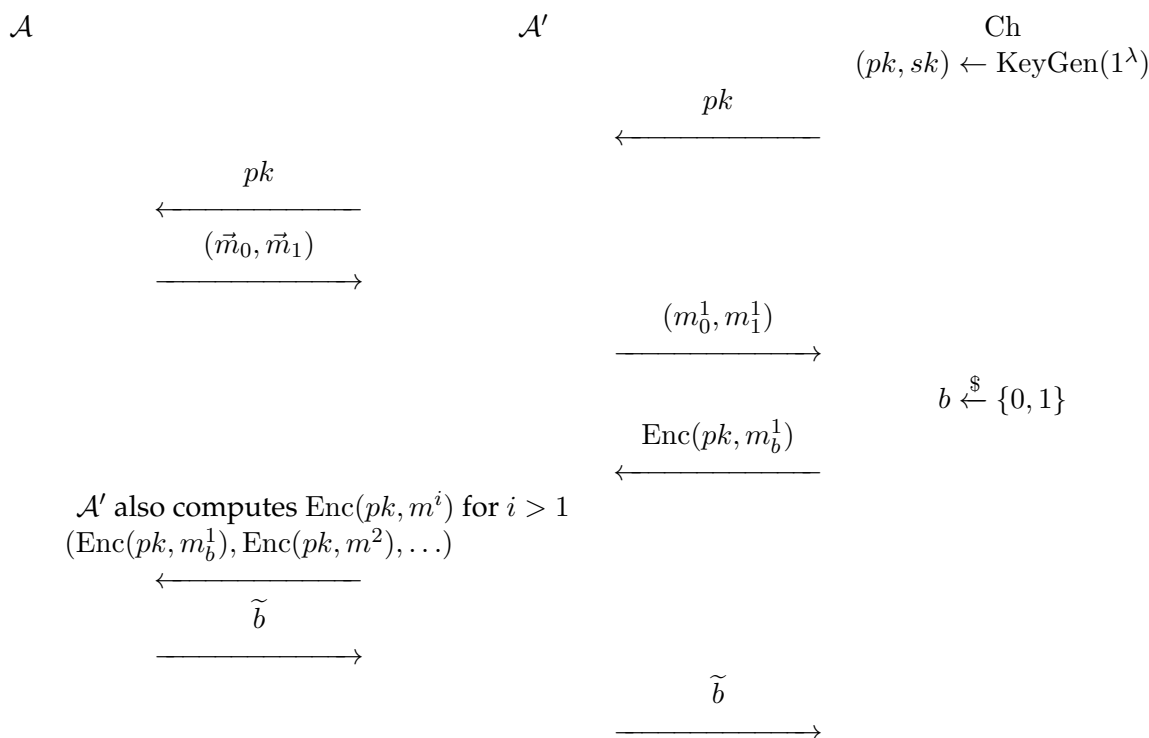
Notice that the security game above involves only a single message pair (m_0, m_1) . For private-key encryption, recall that we distinguished single-message from multi-message security, since the

latter is genuinely stronger (e.g. the OTP satisfies single-message security but not multi-message security). For public-key encryption, the two notions *coincide*: any PKE secure against single-message attacks is also secure against multi-message attacks.

Multi-message security trivially implies single-message security (the latter is a special case with $|\vec{m}| = 1$). The non-trivial direction is the reverse, which we prove by a *hybrid argument*.

18.2.1 The Limited Multi-Message Adversary

We first restrict attention to adversaries whose vectors \vec{m}_0, \vec{m}_1 differ only on the first message. That is, $m_0^1 \neq m_1^1$ but $m_0^i = m_1^i$ for all $i > 1$. Suppose such an adversary \mathcal{A} wins the multi-message game with non-negligible advantage. We construct an adversary \mathcal{A}' for the single-message game that gains the same advantage:



Crucially, \mathcal{A}' can compute $\text{Enc}(pk, m^i)$ for $i > 1$ herself (since she holds the public key), so she sends \mathcal{A} a complete ciphertext vector that matches an encryption of \vec{m}_b . Whatever advantage \mathcal{A} has in distinguishing \vec{m}_0 from \vec{m}_1 becomes the advantage \mathcal{A}' has in distinguishing m_0^1 from m_1^1 .

18.2.2 The Hybrid Argument

To handle adversaries whose vectors differ in arbitrary positions, we use a chain of hybrid games. Let hybrid j be the multi-message game in which Ch encrypts m_1^i for $i \leq j$ and m_0^i for $i > j$.¹⁰ Then:

¹⁰Note: this convention swaps the message vector front-to-back, contrary to one natural choice; either is fine, and this one matches the limited-adversary analysis above.

- Hybrid 0 is the original multi-message game with $b = 0$.
- Hybrid n is the original multi-message game with $b = 1$.
- Hybrids j and $j + 1$ differ only on the $(j + 1)$ -th encryption.

By the limited-adversary argument, no QPT can distinguish hybrids j and $j + 1$ except with negligible advantage. By the triangle inequality, no QPT can distinguish hybrid 0 from hybrid n except with advantage at most $n \cdot \text{negl}(\lambda) = \text{negl}(\lambda)$ (since n is polynomial in λ).

Quantitatively, suppose every QPT \mathcal{A} achieves $|\Pr[\tilde{b} = 0 \mid b = 0] - \Pr[\tilde{b} = 0 \mid b = 1]| \leq 2\mu(\lambda)$ in the single-message game. Then by the hybrid argument, the same difference in the multi-message game is bounded by $2n\mu(\lambda)$. Translating back to the standard “guess” notation:

$$\begin{aligned}
\Pr[\tilde{b} = b] &= \frac{1}{2} \Pr[\tilde{b} = 0 \mid b = 0] + \frac{1}{2} \Pr[\tilde{b} = 1 \mid b = 1] \\
&= \frac{1}{2} \Pr[\tilde{b} = 0 \mid b = 0] + \frac{1}{2}(1 - \Pr[\tilde{b} = 0 \mid b = 1]) \\
&= \frac{1}{2} + \frac{1}{2}(\Pr[\tilde{b} = 0 \mid b = 0] - \Pr[\tilde{b} = 0 \mid b = 1]) \\
&\leq \frac{1}{2} + n\mu(\lambda).
\end{aligned}$$

Since n is polynomial and μ is negligible, $n\mu(\lambda)$ is negligible. Hence single-message security implies multi-message security.

18.3 Regev Encryption

We now construct a public-key encryption scheme for classical bits based on LWE. Parameters: n, m, q (with q prime) and an error distribution χ , all potentially functions of λ .

Scheme.

- $\text{KeyGen}(1^\lambda)$: sample $A \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$, $\vec{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\vec{e} \leftarrow \chi^m$. Output $pk = (A, A\vec{s} + \vec{e})$ and $sk = \vec{s}$.
- $\text{Enc}(pk, \mu \in \{0, 1\})$: parse $pk = (A, \vec{b})$, sample $\vec{r} \xleftarrow{\$} \{0, 1\}^m$. Output $ct = (\vec{r}A, \vec{r}\vec{b} + \mu \lfloor q/2 \rfloor)$.
- $\text{Dec}(sk, ct)$: parse $sk = \vec{s}$ and $ct = (\vec{c}_1, c_2)$. Compute $c_2 - \vec{c}_1\vec{s}$. Output 1 if the result lies in $[q/4, 3q/4] \pmod{q}$, and 0 otherwise.

18.3.1 Correctness

Fix any (pk, sk) in the support of KeyGen , message μ , randomness \vec{r} . The ciphertext components are $\vec{c}_1 = \vec{r}A$ and $c_2 = \vec{r}(A\vec{s} + \vec{e}) + \mu \lfloor q/2 \rfloor$. During decryption,

$$c_2 - \vec{c}_1\vec{s} = \vec{r}(A\vec{s} + \vec{e}) + \mu \lfloor q/2 \rfloor - \vec{r}A\vec{s} = \vec{r}\vec{e} + \mu \lfloor q/2 \rfloor.$$

If $|\vec{r}\vec{e}| \leq q/10$, then $c_2 - \vec{c}_1\vec{s}$ is in $[0, q/10] \cup [9q/10, q]$ if $\mu = 0$, and in $[2q/5, 3q/5]$ if $\mu = 1$. The decryption rule outputs the correct μ .

Bounding $|\vec{r}\vec{e}|$. Since $\vec{r} \in \{0, 1\}^m$, $\vec{r}\vec{e}$ is a sum of at most m entries of \vec{e} . If each entry has absolute value at most β , then $|\vec{r}\vec{e}| \leq m\beta$. Choosing χ supported on values of magnitude $\leq q/(10m)$ guarantees correctness.

18.3.2 Security

We prove security under decisional LWE: no QPT can distinguish $(A, A\vec{s} + \vec{e})$ from (A, \vec{b}) for uniform \vec{b} .

Step 1: Switch to a random public key. Define two games. *Game 0* is the standard security game with $pk = (A, A\vec{s} + \vec{e})$. *Game 1* replaces pk with (A, \vec{b}) for uniform \vec{b} . We show that any QPT changes its $\Pr[\tilde{b} = b]$ between the games by at most $\text{negl}(\lambda)$.

Suppose \mathcal{A} has $\Pr[\tilde{b} = b]$ higher by p' in Game 0 than in Game 1. Construct \mathcal{A}' that breaks DLWE: \mathcal{A}' receives a sample (A, \vec{b}) from the DLWE challenger, embeds it as pk in a simulation of the PKE security game, runs \mathcal{A} , and outputs $\text{guess} = 0$ if $\tilde{b} = b$ and 1 otherwise. The probability of correctly guessing the DLWE bit is

$$\frac{1}{2}p + \frac{1}{2}(1 - (p - p')) = \frac{1}{2} + \frac{p'}{2},$$

where $p = \Pr[\tilde{b} = b]$ in Game 0. By DLWE, $p'/2$ is negligible, so p' is negligible.

Step 2: Game 1 is statistically secure. Let $C = [A \mid \vec{b}] \in \mathbb{Z}_q^{m \times (n+1)}$, a uniformly random matrix. The information \mathcal{A} receives in Game 1 is C and $\vec{r}C + [0, \dots, 0, \mu_b \lfloor q/2 \rfloor]$.

The *Leftover Hash Lemma* [ILL89] states: for $C \xleftarrow{\$} \mathbb{Z}_q^{m \times (n+1)}$, $\vec{r} \xleftarrow{\$} \{0, 1\}^m$, and $\vec{v} \xleftarrow{\$} \mathbb{Z}_q^{n+1}$,

$$(C, \vec{r}C) \approx_s (C, \vec{v}),$$

i.e. $\vec{r}C$ is statistically close to uniform conditional on C , provided m is sufficiently larger than $(n+1) \log q$. Hence the second component of \mathcal{A}' 's view in Game 1 is statistically close to a uniform vector, masking the additive $\mu_b \lfloor q/2 \rfloor$ entirely. In Game 1, no adversary (bounded or otherwise) can guess b with probability exceeding $\frac{1}{2} + \text{negl}(\lambda)$.

Conclusion. By Steps 1 and 2, every QPT \mathcal{A} guesses b with probability $\leq \frac{1}{2} + \text{negl}(\lambda)$ in Game 0. The Regev scheme is therefore secure under decisional LWE.

18.4 Public-Key Encryption of Qubits

We now extend public-key encryption to qubits using the QOTP. Let $(\text{KeyGen}_c, \text{Enc}_c, \text{Dec}_c)$ be any quantum-secure classical PKE.

Construction.

- $\text{KeyGen}_q(1^\lambda)$: run $\text{KeyGen}_c(1^\lambda)$ to get (pk_c, sk_c) . Output $(pk_q = pk_c, sk_q = sk_c)$.
- $\text{Enc}_q(pk_q, \rho)$: sample $a, b \xleftarrow{\$} \{0, 1\}$, compute $\sigma = X^a Z^b \rho (X^a Z^b)^\dagger$. Set $ct_c = \text{Enc}_c(pk_c, (a, b))$. Output $ct_q = (\sigma, ct_c)$.
- $\text{Dec}_q(sk_q, (\sigma, ct_c))$: compute $(a, b) = \text{Dec}_c(sk_c, ct_c)$ and output $(X^a Z^b)^\dagger \sigma (X^a Z^b)$.

The scheme QOTP-masks the qubit with random Pauli keys (a, b) , then classically PKE-encrypts the keys.

18.4.1 Correctness

For any (pk_q, sk_q) and any ρ , by correctness of the classical scheme Dec_c recovers the exact (a, b) used. Substituting:

$$\text{Dec}_q(sk_q, (\sigma, ct_c)) = (X^a Z^b)^\dagger \sigma (X^a Z^b) = (X^a Z^b)^\dagger (X^a Z^b) \rho (X^a Z^b)^\dagger (X^a Z^b) = \rho.$$

18.4.2 Security

We argue via a hybrid argument with four hybrids:

- Hybrid 0: standard security game with Ch encrypting ρ_0 .
- Hybrid 1: identical, but $ct_c = \text{Enc}_c(pk_c, (0, 0))$ regardless of (a, b) .
- Hybrid 2: identical to Hybrid 1, but Ch encrypts ρ_1 instead of ρ_0 .
- Hybrid 3: standard security game with Ch encrypting ρ_1 .

Hybrid 0 \rightarrow Hybrid 1. Any QPT that distinguishes these two with advantage p yields a QPT that breaks the classical scheme with advantage p , by embedding the (unknown) classical ciphertext into the qubit ciphertext. Specifically, an adversary \mathcal{A}' against the classical scheme simulates the qubit security game, asks \mathcal{A} for ρ_0 , samples (a, b) , computes σ and submits to the classical challenger the messages $\mu_0 = (a, b)$ and $\mu_1 = (0, 0)$. The classical challenger returns $\text{Enc}_c(pk_c, \mu_{b'})$, which \mathcal{A}' forwards as ct_c . If $b' = 0$, \mathcal{A}' 's view matches Hybrid 0; if $b' = 1$, it matches Hybrid 1. \mathcal{A}' 's output bit is forwarded as \mathcal{A} 's guess. Hence \mathcal{A}' 's advantage matches \mathcal{A} 's, and is negligible by classical security.

Hybrid 1 \rightarrow Hybrid 2. In both hybrids, (a, b) is uniformly random and the classical ciphertext is $\text{Enc}_c(pk_c, (0, 0))$ — independent of (a, b) . The adversary's view of the quantum component $\sigma = X^a Z^b \rho_{0 \text{ or } 1} (X^a Z^b)^\dagger$, averaged over uniform (a, b) , equals $\frac{\mathbb{I}}{2}$ by the QOTP guarantee, *regardless* of whether the underlying state is ρ_0 or ρ_1 . Hence the two hybrids are *perfectly* indistinguishable.

Hybrid 2 \rightarrow Hybrid 3. Symmetric to Hybrid 0 \rightarrow Hybrid 1.

By the triangle inequality, no QPT distinguishes Hybrid 0 from Hybrid 3 with non-negligible advantage. The PKE for qubits is secure.

Key Takeaways. For public-key encryption, single-message security implies multi-message security via a hybrid argument — a feature absent in the symmetric setting because the simulator can encrypt arbitrary messages on her own using the public key. Regev's scheme builds a single-message-secure PKE from DLWE: keys are LWE samples, encryption uses a random binary mask of the public key, and decryption uses the rounding $\lfloor q/2 \rfloor$ trick. Combining the QOTP with any quantum-secure classical PKE then yields a PKE for arbitrary qubits, with a clean four-hybrid security proof.

19 Homomorphic Encryption for Classical Circuits

In the previous lecture we introduced Regev's public-key encryption scheme, whose security follows from LWE. In this lecture we extend the construction to a *Fully Homomorphic Encryption* (FHE) scheme for classical circuits, due to Gentry, Sahai, and Waters [GSW13] (GSW). FHE will, in turn, be the substrate for FHE over quantum circuits in the following lectures.

Notation. Bold lowercase $\mathbf{a}, \mathbf{b}, \dots$ denote column vectors, $\mathbf{a}^T, \mathbf{b}^T, \dots$ denote row vectors, and bold uppercase $\mathbf{A}, \mathbf{B}, \dots$ denote matrices. $\log x$ denotes base-2 logarithm.

The *Kronecker product* $\mathbf{A} \otimes \mathbf{B}$ replaces each entry a_{ij} of \mathbf{A} with the block $a_{ij}\mathbf{B}$.

19.1 Regev Public-Key Encryption Recap

Recall the public-key encryption scheme from the last lecture, with slight notational adjustment for compatibility with the GSW construction:

- $\text{KeyGen}(1^\lambda)$:
 - Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^{n \times 1}$, $\mathbf{e} \leftarrow \chi_B^{m \times 1}$, where $m = (n+1)\lceil \log q \rceil$, and χ is a bounded error distribution with bound $B < \frac{1}{m} \lfloor \frac{q}{4} \rfloor$.
 - Set $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^{m \times 1}$ and $\mathbf{B} = [\mathbf{A} \mid \mathbf{b}] \in \mathbb{Z}_q^{m \times (n+1)}$.
 - Output $pk = \mathbf{B}$ and $sk = \mathbf{t} = \begin{bmatrix} -\mathbf{s} \\ 1 \end{bmatrix} \in \mathbb{Z}_q^{(n+1) \times 1}$.

- $\text{Enc}(pk = \mathbf{B}, \mu)$: sample $\mathbf{r} \in \{0, 1\}^{m \times 1}$. Output

$$\mathbf{c}^T = \mathbf{r}^T \mathbf{B} + [0, \dots, 0, \mu \lfloor \frac{q}{2} \rfloor] \in \mathbb{Z}_q^{1 \times (n+1)}.$$

- $\text{Dec}(sk = \mathbf{t}, \mathbf{c}^T)$: compute $c = \mathbf{c}^T \mathbf{t} \in \mathbb{Z}_q$. Output 1 if $\lfloor \frac{q}{4} \rfloor < c < \lfloor \frac{3q}{4} \rfloor \pmod{q}$, else 0.

Correctness and security follow as discussed in the last lecture. Observe that $\mathbf{B}\mathbf{t} = \mathbf{e}$: writing out $\mathbf{B}\mathbf{t} = [\mathbf{A} \mid \mathbf{b}] \begin{bmatrix} -\mathbf{s} \\ 1 \end{bmatrix} = -\mathbf{A}\mathbf{s} + \mathbf{b} = -\mathbf{A}\mathbf{s} + \mathbf{A}\mathbf{s} + \mathbf{e} = \mathbf{e}$. So during decryption,

$$\mathbf{c}^T \mathbf{t} = \mathbf{r}^T \mathbf{B}\mathbf{t} + \mu \lfloor \frac{q}{2} \rfloor = \mathbf{r}^T \mathbf{e} + \mu \lfloor \frac{q}{2} \rfloor,$$

which is a small noise plus $\mu \lfloor \frac{q}{2} \rfloor$.

XOR Homomorphism. Given two ciphertexts $\mathbf{c}_1^T, \mathbf{c}_2^T$ encrypting μ_1, μ_2 :

$$\begin{aligned}\mathbf{c}_1^T &= \mathbf{r}_1^T \mathbf{B} + [0, \dots, 0, \mu_1 \lfloor \frac{q}{2} \rfloor], \\ \mathbf{c}_2^T &= \mathbf{r}_2^T \mathbf{B} + [0, \dots, 0, \mu_2 \lfloor \frac{q}{2} \rfloor].\end{aligned}$$

Their sum is

$$\mathbf{c}_1^T + \mathbf{c}_2^T = (\mathbf{r}_1^T + \mathbf{r}_2^T) \mathbf{B} + [0, \dots, 0, (\mu_1 + \mu_2) \lfloor \frac{q}{2} \rfloor].$$

This is a valid ciphertext encrypting $(\mu_1 + \mu_2) \bmod 2 = \mu_1 \oplus \mu_2$, with a slightly larger noise term.

Failure of AND Homomorphism. We claim the Regev scheme does *not* support AND homomorphism: were both XOR and AND available, we would have FHE immediately (since XOR and AND are complete for Boolean circuits). Consider the matrix product $\mathbf{c}_1^T \mathbf{c}_2$ for ciphertexts as above. Even ignoring dimensional mismatches, decryption would require $\mathbf{c}_1^T \mathbf{c}_2$. The right factor \mathbf{c}_2 has small norm plus $\mu_2 \lfloor \frac{q}{2} \rfloor$; left-multiplying by $\mathbf{c}_1^T = \mathbf{r}_1^T \mathbf{B} + [0, \dots, 0, \mu_1 \lfloor \frac{q}{2} \rfloor]$, the dominant term $\mathbf{r}_1^T \mathbf{B}$ has large norm and multiplies the small noise of \mathbf{c}_2 , blowing it up beyond the decryption threshold $q/4$. The naive product is non-decryptable.

19.2 The GSW FHE Scheme

The GSW scheme [GSW13, Hal17] works around the noise-blowup problem by *bit-decomposing* ciphertexts before multiplication. Bit-decomposition replaces a high-norm matrix with a binary matrix, which when multiplied by the other ciphertext's noise term yields only a polynomially-larger noise.

Gadget Matrix. Let q be a positive integer modulus and $\ell = \lceil \log q \rceil$. Define

$$\mathbf{g}^T = [1, 2, 2^2, \dots, 2^{\ell-1}] \in \mathbb{Z}_q^{1 \times \ell}.$$

The bit-decomposition operation $\mathbf{g}^{-1}: \mathbb{Z}_q \rightarrow \{0, 1\}^\ell$ takes $x \in \mathbb{Z}_q$ to its binary representation, so that $\mathbf{g}^T \mathbf{g}^{-1}(x) = x$.

For $n \geq 1$, define the *gadget matrix*

$$\mathbf{G}_n = \mathbb{I}_{n \times n} \otimes \mathbf{g} \in \mathbb{Z}_q^{n\ell \times n}.$$

The bit-decomposition extends entry-wise to matrices: $\mathbf{G}^{-1}: \mathbb{Z}_q^{m \times n} \rightarrow \{0, 1\}^{m \times n\ell}$. The key identity is

$$\mathbf{G}^{-1}(\mathbf{A}) \cdot \mathbf{G} = \mathbf{A}. \quad (19.1)$$

Note that \mathbf{G}^{-1} is an *operation* (returning a wider matrix), while \mathbf{G} is a fixed *matrix*.

Scheme.

- $\text{KeyGen}(1^\lambda)$:
 - Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^{n \times 1}$, $\mathbf{e} \leftarrow \chi_B^{m \times 1}$, with $m = (n + 1) \lceil \log q \rceil$ and B chosen depending on the homomorphic-operation budget.

- Set $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$ and $\mathbf{B} = [\mathbf{A} \mid \mathbf{b}]$. Output $pk = \mathbf{B}$ and $sk = \mathbf{t} = \begin{bmatrix} -\mathbf{s} \\ 1 \end{bmatrix}$.

• $\text{Enc}(pk = \mathbf{B}, \mu)$: sample $\mathbf{R} \in \{0, 1\}^{m \times m}$. Output

$$\mathbf{C} = \mathbf{R}\mathbf{B} + \mu\mathbf{G}_{n+1} \in \mathbb{Z}_q^{m \times (n+1)}.$$

• $\text{Dec}(sk = \mathbf{t}, \mathbf{C})$:

- Let $\mathbf{w}^T = [0, \dots, 0, \lfloor \frac{q}{2} \rfloor] \in \mathbb{Z}_q^{1 \times (n+1)}$.
- Compute $c = \mathbf{G}^{-1}(\mathbf{w}^T) \cdot \mathbf{C} \cdot \mathbf{t}$.
- Output 1 if $\lfloor \frac{q}{4} \rfloor < c < \lfloor \frac{3q}{4} \rfloor$, else 0.

Correctness. For $\mathbf{C} = \mathbf{R}\mathbf{B} + \mu\mathbf{G}_{n+1}$,

$$\begin{aligned} \mathbf{G}^{-1}(\mathbf{w}^T) \cdot \mathbf{C} \cdot \mathbf{t} &= \mathbf{G}^{-1}(\mathbf{w}^T) \cdot (\mathbf{R}\mathbf{B} + \mu\mathbf{G}_{n+1}) \cdot \mathbf{t} \\ &= \mathbf{G}^{-1}(\mathbf{w}^T) \cdot \mathbf{R}\mathbf{B}\mathbf{t} + \mu\mathbf{G}^{-1}(\mathbf{w}^T) \cdot \mathbf{G}_{n+1}\mathbf{t} \\ &= \mathbf{G}^{-1}(\mathbf{w}^T) \cdot \mathbf{R}\mathbf{e} + \mu\mathbf{w}^T\mathbf{t} \\ &= \mathbf{G}^{-1}(\mathbf{w}^T) \cdot \mathbf{R}\mathbf{e} + \mu\left\lfloor \frac{q}{2} \right\rfloor, \end{aligned}$$

where we used $\mathbf{B}\mathbf{t} = \mathbf{e}$ and (19.1). Both $\mathbf{G}^{-1}(\mathbf{w}^T)$ and \mathbf{R} are binary matrices, so $|\mathbf{G}^{-1}(\mathbf{w}^T) \cdot \mathbf{R} \cdot \mathbf{e}|$ remains a controlled-size noise. Choosing the initial error bound B small enough to keep this noise below $q/4$ ensures correct decryption.

Security. Security follows by the same DLWE argument as for Regev encryption.

XOR Homomorphism. For ciphertexts $\mathbf{C}_i = \mathbf{R}_i\mathbf{B} + \mu_i\mathbf{G}_{n+1}$ ($i = 1, 2$),

$$\mathbf{C}_1 + \mathbf{C}_2 = (\mathbf{R}_1 + \mathbf{R}_2)\mathbf{B} + (\mu_1 + \mu_2)\mathbf{G}_{n+1}.$$

Decryption returns $(\mu_1 + \mu_2) \bmod 2 = \mu_1 \oplus \mu_2$. The noise term doubles but remains bounded.

AND Homomorphism. Define

$$\mathbf{C}' = \mathbf{G}^{-1}(\mathbf{C}_1) \cdot \mathbf{C}_2.$$

We claim \mathbf{C}' is a valid encryption of $\mu_1\mu_2$.

Decryption computes $\mathbf{C}'\mathbf{t}$:

$$\begin{aligned} \mathbf{G}^{-1}(\mathbf{C}_1) \cdot \mathbf{C}_2 \cdot \mathbf{t} &= \mathbf{G}^{-1}(\mathbf{C}_1) \cdot (\mathbf{R}_2\mathbf{B}\mathbf{t} + \mu_2\mathbf{G}_{n+1}\mathbf{t}) \\ &= \mathbf{G}^{-1}(\mathbf{C}_1) \cdot \mathbf{R}_2\mathbf{e} + \mu_2\mathbf{G}^{-1}(\mathbf{C}_1) \cdot \mathbf{G}_{n+1}\mathbf{t} \\ &\stackrel{(19.1)}{=} \mathbf{G}^{-1}(\mathbf{C}_1) \cdot \mathbf{R}_2\mathbf{e} + \mu_2\mathbf{C}_1\mathbf{t} \\ &= \mathbf{G}^{-1}(\mathbf{C}_1) \cdot \mathbf{R}_2\mathbf{e} + \mu_2(\mathbf{R}_1\mathbf{e} + \mu_1\mathbf{G}_{n+1}\mathbf{t}) \\ &= (\mathbf{G}^{-1}(\mathbf{C}_1)\mathbf{R}_2 + \mu_2\mathbf{R}_1)\mathbf{e} + \mu_1\mu_2\mathbf{G}_{n+1}\mathbf{t}. \end{aligned}$$

The decryptor finally applies $\mathbf{G}^{-1}(\mathbf{w}^T)$ on the left as before. The noise factor $\mathbf{G}^{-1}(\mathbf{C}_1)\mathbf{R}_2 + \mu_2\mathbf{R}_1$ is a sum of binary matrices, so the noise is at most multiplied by $O(m)$ rather than blowing up exponentially. Choosing the initial error bound to accommodate the noise growth over the required number of AND gates gives correct decryption.

Discussion. The GSW scheme is homomorphic with respect to both XOR and AND, hence (for circuits of any bounded depth d) supports any classical circuit. Strictly speaking, this is *levelled FHE*: the parameters depend on a circuit depth bound d fixed at key-generation time. To obtain full FHE without depth limits, one applies Gentry's *bootstrapping* technique on top — but this is beyond our scope.

Key Takeaways. Regev encryption is additively homomorphic but not multiplicatively. The GSW scheme augments Regev with a *gadget-matrix* representation: ciphertexts become wider matrices, and the bit-decomposition operation \mathbf{G}^{-1} converts a high-norm ciphertext into a binary matrix before multiplication. This keeps the noise blow-up to a polynomial factor per AND gate, yielding (levelled) FHE under the LWE assumption.

20 Homomorphic Encryption for Quantum Circuits

In this lecture, we begin our study of homomorphic encryption for quantum circuits. Towards this, we first describe a (relatively easy) construction supporting only Clifford gates, and then identify the technical step required to extend it to a fully homomorphic encryption (FHE) scheme over quantum circuits. The full extension uses *Trapdoor Claw-Free Functions* (TCFs), which we introduce here.

20.1 Quantum Homomorphic Encryption: Definition

Definition 24. A quantum homomorphic encryption scheme is a tuple $(\text{KeyGen}, \text{Enc}, \text{Eval}, \text{Dec})$ such that:

- $\text{KeyGen}(1^\lambda) \rightarrow (pk, sk)$ generates a public-key/secret-key pair.
- $\text{Enc}(pk, \rho) \rightarrow ct$ takes a quantum state ρ and outputs a ciphertext.
- $\text{Eval}(pk, ct, C) \rightarrow ct'$ takes a ciphertext and a quantum circuit C and outputs a transformed ciphertext.
- $\text{Dec}(sk, ct') \rightarrow \rho'$ takes a transformed ciphertext and outputs a quantum state.

The scheme is *correct* if $\text{Dec}(sk, \text{Eval}(pk, \text{Enc}(pk, \rho), C)) \approx C(\rho)$ with high probability, and *secure* if $(pk, \text{Enc}(pk, \rho_0))$ is computationally indistinguishable from $(pk, \text{Enc}(pk, \rho_1))$ for any two states ρ_0, ρ_1 of the same dimension.

We restrict attention to quantum circuits over the universal gate set $\{H, X, Z, \text{CNOT}, T, \text{Toffoli}\}$ for concreteness; recall that the first five form the Clifford group, and Toffoli (or equivalently T) lifts it to BQP-complete.

20.2 A Scheme for Clifford Circuits

20.2.1 Setup

We start from the public-key encryption scheme for qubits from Lecture 18: given a classical PKE $(\text{Gen}_c, \text{Enc}_c, \text{Dec}_c)$ that is quantum-secure (e.g. GSW), we encrypt a qubit $|\psi\rangle$ by sampling $a, b \xleftarrow{\$} \{0, 1\}$ and outputting

$$ct = (X^a Z^b |\psi\rangle, \text{Enc}_c(pk_c, (a, b))).$$

For multi-qubit states ρ on n qubits, sample (a_i, b_i) for each $i \in [n]$ and apply $X^{a_i} Z^{b_i}$ qubitwise:

$$ct = ((X^{a_1} Z^{b_1} \otimes \dots \otimes X^{a_n} Z^{b_n}) \rho (X^{a_1} Z^{b_1} \otimes \dots \otimes X^{a_n} Z^{b_n})^\dagger, \text{Enc}_c(pk_c, (a_1, b_1, \dots, a_n, b_n))).$$

20.2.2 Homomorphic Evaluation: Single-Qubit Cliffords

Consider applying a single-qubit Clifford gate C to the QOTP-encrypted qubit $X^a Z^b |\psi\rangle$. We compute

$$C X^a Z^b |\psi\rangle.$$

Because C normalizes the Pauli group — i.e. for every Clifford C and Pauli P , CPC^\dagger is another Pauli (up to phase) — there exists a Pauli P' with $CX^a Z^b = P'C$. Concretely, $P' = C(X^a Z^b)C^\dagger$, and so

$$C X^a Z^b |\psi\rangle = P' C |\psi\rangle.$$

Thus applying C to the quantum component transforms the ciphertext to one masked by a new Pauli P' . The Pauli keys (a, b) are encrypted classically; using the classical scheme's homomorphic operations, we can update the encrypted key from (a, b) to (a', b') where $P' = X^{a'} Z^{b'}$.

For example:

- $X \cdot (X^a Z^b) = X^{a+1} Z^b$, so $(a, b) \rightarrow (a + 1, b)$.
- $Z \cdot (X^a Z^b) = (-1)^a X^a Z^{b+1}$ (global phase ignored), so $(a, b) \rightarrow (a, b + 1)$.
- $H \cdot (X^a Z^b) = (-1)^{ab} Z^a X^b H = (-1)^{ab} X^b Z^a H$, so $(a, b) \rightarrow (b, a)$.

20.2.3 Two-Qubit Cliffords: CNOT

For the CNOT gate acting on two QOTP-encrypted qubits with masks (a_1, b_1) and (a_2, b_2) :

$$\text{CNOT} \cdot (X^{a_1} Z^{b_1} \otimes X^{a_2} Z^{b_2}) = (X^{a_1} Z^{b_1+b_2} \otimes X^{a_1+a_2} Z^{b_2}) \cdot \text{CNOT}.$$

So applying CNOT and updating the encrypted keys to $(a_1, b_1 + b_2, a_1 + a_2, b_2)$ gives a correctly QOTP-encrypted output.

20.2.4 Limitation: Non-Clifford Gates

The scheme thus far evaluates any *Clifford* circuit homomorphically. To support universal quantum computation we need a non-Clifford gate: the Toffoli or the T gate. Either suffices, and we focus on Toffoli for concreteness.

Why Toffoli is hard. The action of Toffoli on QOTP-encrypted inputs does not simply produce a Toffoli with updated Pauli masks; instead, it produces a Toffoli plus a leftover entanglement with the encrypted keys. Specifically, one can show that

$$\text{Toffoli} \cdot (X^{a_1} Z^{b_1} \otimes X^{a_2} Z^{b_2} \otimes X^{a_3} Z^{b_3}) = (\text{Pauli mask}) \cdot \text{CNOT}^{a_1 b_2 \cdot \text{stuff}} \dots \text{Toffoli},$$

where the leftover CNOTs are *controlled by the encrypted Pauli keys*. To make this work, we need an *encrypted CNOT*: a CNOT whose control bit is encrypted under the classical FHE.

20.3 Encrypted CNOT

Goal. Given $\text{Enc}_c(s)$ (a classical encryption of a single bit s) and a two-qubit state $|\psi\rangle$, perform $\text{CNOT}^s |\psi\rangle$ homomorphically — producing a QOTP-encrypted output whose Pauli keys are themselves classically encrypted, so the holder of the FHE secret key can decrypt and obtain $\text{CNOT}^s |\psi\rangle$.

The construction proceeds by:

1. Convert $\text{Enc}_c(s)$ into a *TCF encryption* of s (a TCF pair whose hidden bit is s).
2. Use the TCF encryption to entangle a claw with $|\psi\rangle$, then measure to leave the CNOT outcome under a QOTP mask whose classical keys can be FHE-evaluated using the trapdoor.

We treat Step 2 as a black box for this lecture and concentrate on Step 1, namely the introduction and construction of TCFs.

20.4 Trapdoor Claw-Free Functions

Definition 25 (TCF Pair). A *Trapdoor Claw-Free Function pair* consists of:

- Two functions $f_0, f_1: \mathcal{X} \rightarrow \mathcal{Y}$, with the same image, each injective. We refer to a pair (x_0, x_1) with $f_0(x_0) = f_1(x_1)$ as a *claw*.
- A trapdoor td that allows efficient inversion: $\text{Invert}(td, y) = (x_0, x_1)$ recovers the claw at y .

Required properties:

1. *Claw-freeness.* Without td , no QPT can find a claw except with negligible probability.
2. *Hardcore bit / adaptive hardcore property.* There is a designated bit s associated with each TCF pair (the “hidden bit”) such that, for every claw (x_0, x_1) , $x_0[1] \oplus x_1[1] = s$ (where $x[1]$ denotes the first bit of x).

The adaptive hardcore property implies the TCF pair functions as a kind of encryption of s : any party that could efficiently extract s from (f_0, f_1) would also be able to compute claws.

TCFs from LWE. Let Enc_{FHE} be a quantum-secure (levelled) FHE scheme with public/secret keys (pk, sk) . Given a classical FHE encryption $\text{Enc}_{\text{FHE}}(s; r_s)$ of the bit s , we construct the TCF pair as follows:

$$\begin{aligned} f_0(x_0) &= \text{Enc}_{\text{FHE}}(x_0[1]; r_{x_0}), \\ f_1(x_1) &= f_0(x_1) \oplus \text{Enc}_{\text{FHE}}(s; r_s), \end{aligned}$$

where $x_0[1]$ is the first bit of x_0 and the remaining bits of x_0 serve as the encryption randomness r_{x_0} . (Similarly for x_1 .) The trapdoor is the FHE secret key sk .

Adaptive hardcore property. A claw (x_0, x_1) satisfies $f_0(x_0) = f_1(x_1)$, i.e.

$$\text{Enc}_{\text{FHE}}(x_0[1]; r_{x_0}) = \text{Enc}_{\text{FHE}}(x_1[1]; r_{x_1}) \oplus \text{Enc}_{\text{FHE}}(s; r_s).$$

Decrypting both sides (which is what the trapdoor enables, but is also what any decryptor sees in the “plaintext space”), we get $x_0[1] = x_1[1] \oplus s$, equivalently $x_0[1] \oplus x_1[1] = s$.

Claw-freeness. Suppose an adversary could find a claw (x_0, x_1) in polynomial time. By the adaptive hardcore property, the adversary could then determine s from the TCF pair, contradicting the semantic security of $\text{Enc}_{\text{FHE}}(s)$.

20.5 The Plan for Quantum FHE

With TCFs available, we can build the encrypted CNOT — which is the central new step in the GSW-style quantum FHE construction. Here is the plan in outline:

1. Start with a quantum input QOTP-masked under FHE-encrypted Pauli keys.
2. For each Toffoli (or T) gate in the circuit, decompose it into Clifford gates plus an encrypted CNOT operation.
3. Run the Clifford gates as in Section 20.2, propagating the Pauli mask homomorphically.
4. Run the encrypted CNOT by (i) converting the relevant FHE-encrypted control bit into a TCF encryption, then (ii) using the TCF encryption to evaluate the CNOT.
5. At the end, the QOTP-masked output is recovered, with all Pauli keys still FHE-encrypted.

We work through the encrypted CNOT construction in Lectures 21 and 22; we have already worked through the latter in the polished version of Lecture 22 above (which deals with the QFT step inside the encrypted CNOT).

Key Takeaways. A quantum FHE scheme starts from the (classical) FHE encryption of QOTP keys and homomorphically lifts each Clifford gate by translating it through the Clifford-Pauli commutation rule and updating the encrypted keys accordingly. Toffoli requires more: a CNOT controlled by an FHE-encrypted bit, which is the central new primitive. To build it we will use a Trapdoor Claw-Free Function pair, in which the difference $x_0[1] \oplus x_1[1]$ across any claw encodes the encrypted control bit. We close out the construction in subsequent lectures.

20.6 Further Reading

- Urmila Mahadev. *Classical Homomorphic Encryption for Quantum Circuits*. [arXiv:1708.02130](https://arxiv.org/abs/1708.02130).
- Brakerski, Christiano, Mahadev, Vazirani, Vidick. *A Cryptographic Test of Quantumness and Certifiable Randomness from a Single Quantum Device*. [BCM⁺18].

21 Quantum Fully Homomorphic Encryption – II

This lecture continues our discussion of quantum FHE. We left off with a scheme that handles Clifford gates by propagating updated Pauli masks through encrypted-key updates. We now address the non-Clifford gate (Toffoli), reducing its homomorphic evaluation to an *encrypted CNOT* step, which we sketch here and complete in the next lecture.

21.1 The Toffoli Gate

Recall the Toffoli (CCNOT) gate acts as $\text{Toffoli} |a, b, c\rangle = |a, b, c \oplus ab\rangle$. We want to evaluate Toffoli on three QOTP-encrypted qubits with Pauli masks (a_i, b_i) for $i = 1, 2, 3$.

Pauli propagation through Toffoli. Toffoli does *not* normalize the Pauli group (a fact one can verify by computing $\text{Toffoli} \cdot X_1 \cdot \text{Toffoli}^\dagger$ and finding it equals X_1 times an additional CNOT). Specifically,

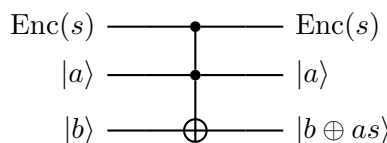
$$\text{Toffoli} \cdot (X^{a_1} \otimes X^{a_2} \otimes X^{a_3}) = (X^{a_1} \otimes X^{a_2} \otimes X^{a_3+a_1a_2}) \cdot \text{CNOT}_{2,3}^{a_1a_2} \cdot \text{CNOT}_{1,3}^{a_1a_2} \dots \text{Toffoli},$$

where the exact form of the leftover CNOTs need not concern us. The essential point is that after Pauli normalization, there remain CNOT gates whose application is controlled by classical values that, in our context, are FHE-encrypted (namely a_1, a_2). These are precisely the *encrypted CNOTs* we need to implement.

21.2 Encrypted CNOT: Setup

Goal. Given the classical FHE encryption $\text{Enc}(s)$ of a bit $s \in \{0, 1\}$ and a two-qubit state $|\psi\rangle$ on registers (A, B) , produce an output that is:

- A two-qubit state on registers (A, B) that equals $\text{CNOT}^s |\psi\rangle$ up to a known QOTP mask (i.e. Pauli matrices applied per qubit);
- Together with FHE encryptions of the Pauli mask, allowing the holder of the FHE secret key to recover $\text{CNOT}^s |\psi\rangle$.



The output is *not* simply $\text{CNOT}^s |\psi\rangle$: that would leak s on an input like $a = 1, b = 0$, which would yield $|a = 1, b = s\rangle$. The QOTP mask is essential.

21.3 From FHE to TCFs

We follow the plan from Lecture 20:

1. Convert $\text{Enc}(s)$ to a TCF encryption of s : a pair (f_0, f_1) such that any claw (x_0, x_1) satisfies $x_0[1] \oplus x_1[1] = s$, and the trapdoor (FHE secret key) allows efficient computation of FHE encryptions of x_0, x_1 from the trapdoor-inverted ciphertext y .
2. Use the TCF pair (f_0, f_1) , plus quantum ancillas, to entangle a claw with $|\psi\rangle$.
3. Measure judiciously to leave the desired CNOT-applied state under a QOTP mask, with the Pauli keys computable as classical functions of (x_0, x_1, d) for a known d .

21.3.1 TCF Construction Recap

For completeness, the TCF construction from Lecture 20:

$$\begin{aligned} f_0(x_0) &= \text{Enc}_{\text{FHE}}(x_0[1]; r_{x_0}), \\ f_1(x_1) &= f_0(x_1) \oplus \text{Enc}_{\text{FHE}}(s; r_s). \end{aligned}$$

Trapdoor: the FHE secret key sk . Claw-freeness: any claw determines s , but s is encrypted under FHE, so no QPT can find claws (otherwise it would break FHE security).

21.4 Entangling a Claw With $|\psi\rangle$

We now sketch the second step of the encrypted-CNOT construction. The detailed analysis is in Lecture 22.

Append two ancilla registers, initialized to a uniform superposition over $x \in \{0, 1\}^n$ in the third register and $|0\rangle$ in the fourth register:

$$|\psi\rangle_{AB} \otimes \sum_x |x\rangle_C \otimes |0\rangle_D = \sum_{a,b,x} \alpha_{a,b} |a, b, x, 0\rangle.$$

Apply the unitary U_{f_0, f_1} that maps $|a, b, x, 0\rangle \mapsto |a, b, x, f_a(x)\rangle$:

$$\sum_{a,b,x} \alpha_{a,b} |a, b, x, f_a(x)\rangle.$$

Measure register D , obtaining some y in the common image of f_0, f_1 . Since f_a is injective, this collapses register C to either $x_0 = f_0^{-1}(y)$ (if the value of register A is 0) or $x_1 = f_1^{-1}(y)$ (if the value of register A is 1):

$$\sum_b \alpha_{0,b} |0, b, x_0\rangle + \sum_b \alpha_{1,b} |1, b, x_1\rangle.$$

21.4.1 Splitting the Claw

Recall that $x_0[1] \oplus x_1[1] = s$ from the adaptive hardcore property. Let $r_a = x_a[2 : n]$ denote the last $n - 1$ bits of x_a . Then

$$\begin{aligned} \sum_b \alpha_{0,b} |0, b, x_0[1], r_0\rangle + \sum_b \alpha_{1,b} |1, b, x_1[1], r_1\rangle &= \sum_b \alpha_{0,b} |0, b, x_0[1], r_0\rangle + \sum_b \alpha_{1,b} |1, b, x_0[1] \oplus s, r_1\rangle \\ &= \sum_{a,b} \alpha_{a,b} |a, b, x_0[1] \oplus as, r_a\rangle, \end{aligned}$$

where the second equality combines the two cases $a = 0, 1$.

CNOTing the third qubit (carrying $x_0[1] \oplus as$) onto the second register (carrying b):

$$\sum_{a,b} \alpha_{a,b} |a, b \oplus x_0[1] \oplus as, x_0[1], r_a\rangle.$$

The third and fourth registers together carry the string x_a , so this can be rewritten as

$$(\mathbb{I} \otimes X^{x_0[1]} \otimes \mathbb{I}) \sum_{a,b} \alpha_{a,b} |a, b \oplus as, x_a\rangle.$$

21.4.2 Removing the Residue via QFT

We are left with the desired state $\sum_{a,b} \alpha_{a,b} |a, b \oplus as\rangle$ tensored with a register carrying $|x_a\rangle$. To unentangle the residue, apply $H^{\otimes n}$ (equivalently, QFT over \mathbb{F}_2^n) to the last register:

$$H^{\otimes n} \sum_{a,b} \alpha_{a,b} |a, b \oplus as, x_a\rangle = \sum_{a,b,d} \alpha_{a,b} (-1)^{x_a \cdot d} |a, b \oplus as, d\rangle.$$

Measure the last register to obtain d . The state collapses to

$$\sum_{a,b} \alpha_{a,b} (-1)^{x_a \cdot d} |a, b \oplus as\rangle.$$

21.4.3 Reading Off the Pauli Mask

We can rewrite $(-1)^{x_a \cdot d} = (-1)^{x_0 \cdot d} \cdot (-1)^{a(x_0 \oplus x_1) \cdot d}$, which gives (up to a global phase $(-1)^{x_0 \cdot d}$):

$$\sum_{a,b} \alpha_{a,b} (-1)^{a(x_0 \oplus x_1) \cdot d} |a, b \oplus as\rangle = (Z^{(x_0 \oplus x_1) \cdot d} \otimes \mathbb{I}) \sum_{a,b} \alpha_{a,b} |a, b \oplus as\rangle.$$

This is the CNOT outcome under a Z mask on the first qubit. Combining with the $X^{x_0[1]}$ mask on the second qubit from the previous subsection,

$$(Z^{(x_0 \oplus x_1) \cdot d} \otimes X^{x_0[1]}) \text{CNOT}^s |\psi\rangle.$$

The Pauli keys $z = (x_0 \oplus x_1) \cdot d$ and $x = x_0[1]$ are computable from (x_0, x_1, d) . Using the trapdoor we can compute FHE encryptions of x_0, x_1 , and from these (together with the public d), FHE encryptions of z and x . The encrypted CNOT is complete.

21.5 Putting It All Together

The full quantum FHE scheme follows by combining:

- The classical FHE scheme (e.g. GSW) to encrypt the Pauli keys;
- The Clifford-evaluation rules from Lecture 20 to homomorphically apply Clifford gates;
- The encrypted-CNOT construction above (and Lecture 22's detailed analysis) to evaluate non-Clifford gates like Toffoli.

The resulting scheme supports any quantum circuit (provided we use a levelled FHE with sufficient circuit-depth budget).

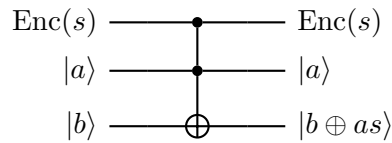
Key Takeaways. The Toffoli gate fails to normalize the Pauli group, leaving behind “leftover CNOTs” whose application is controlled by FHE-encrypted bits. Implementing each such encrypted CNOT requires a Trapdoor Claw-Free Function pair encoding the encrypted control bit; entangling a TCF claw with the quantum input, splitting the claw using the adaptive hardcore property, and applying a QFT followed by a measurement leaves the desired CNOT outcome under a QOTP mask whose keys are classically computable using the trapdoor. We work out the algebra in greater detail in Lecture 22.

21.6 Further Reading

- Urmila Mahadev. *Classical Homomorphic Encryption for Quantum Circuits*. [arXiv:1708.02130](https://arxiv.org/abs/1708.02130).

22 Homomorphic Encryption III

In the previous lecture we showed that homomorphically evaluating a Toffoli gate on QOTP-encrypted data reduces to implementing an *encrypted CNOT*: a CNOT in which the control bit is itself encrypted under a classical homomorphic encryption (HE) scheme. The corresponding gadget is



Given an HE encryption $\text{Enc}(s)$ of a classical bit s and a quantum state $|\psi\rangle$ on two qubits (a, b) , the encrypted CNOT should produce a QOTP-encryption of the standard CNOT outcome — not the bare outcome, since otherwise setting $a = 1$ and $b = 0$ would expose s . The output therefore has the form $X^x Z^z \text{CNOT}^s |\psi\rangle$ for some classical-encrypted Pauli keys (x, z) .

The construction proceeds in two steps:

1. Convert the classical HE-encryption $\text{Enc}(s)$ into a *trapdoor-claw-free function (TCF) encryption* $\text{TCF-Enc}(s)$.
2. Use $\text{TCF-Enc}(s)$ to implement the encrypted CNOT on $|\psi\rangle$, obtaining the desired QOTP-encrypted output.

We address Step 2 first, then Step 1.

22.1 Step 1: TCF Encryption

We first recall the definition of a trapdoor-claw-free function pair.

Definition 26 (Trapdoor claw-free function pair). A *TCF pair* is a tuple (f_0, f_1, td) such that:

1. f_0 and f_1 are injective with the same image;
2. it is computationally hard to find a *claw* (x_0, x_1) with $f_0(x_0) = f_1(x_1)$;
3. there is an efficient trapdoor algorithm $\text{Invert}(td, y) = (x_0, x_1)$ that, on input y in the image, returns the unique claw.

Extra (“adaptive hardcore”) property: there is a hidden bit s associated with (f_0, f_1) such that for every claw, $x_0[1] \oplus x_1[1] = s$, where $x[1]$ denotes the first bit of x .

Because every claw determines s via $s = x_0[1] \oplus x_1[1]$, the TCF pair itself functions as an encryption of s : any party that could efficiently compute s from (f_0, f_1) alone would also be able to find claws.

22.2 Step 2: Implementing Encrypted CNOT from TCF-Enc(s)

Given the TCF pair encrypting s and a state $|\psi\rangle = \sum_{a,b} \alpha_{a,b} |a, b\rangle$, we implement

$$\sum_{a,b} \alpha_{a,b} |a, b\rangle \longrightarrow \text{Enc}\left(\sum_{a,b} \alpha_{a,b} |a, b \oplus as\rangle\right),$$

where the encryption on the right is a QOTP with Pauli keys to be determined.

Entangling a claw with $|\psi\rangle$. Append two ancilla registers initialized to a uniform superposition over $x \in \{0, 1\}^n$ and to $|0\rangle$:

$$|\psi\rangle \otimes \sum_x |x\rangle \otimes |0\rangle = \sum_{a,b,x} \alpha_{a,b} |a, b, x, 0\rangle.$$

Apply the unitary U_{f_0, f_1} that maps $|a, b, x, 0\rangle \mapsto |a, b, x, f_a(x)\rangle$:

$$\sum_{a,b,x} \alpha_{a,b} |a, b, x, f_a(x)\rangle.$$

Measure the last register, obtaining some y in the common image. Since f_a is injective, this collapses the third register to either $x_0 = f_0^{-1}(y)$ or $x_1 = f_1^{-1}(y)$ depending on the first register:

$$\sum_b \alpha_{0,b} |0, b, x_0\rangle + \sum_b \alpha_{1,b} |1, b, x_1\rangle.$$

Splitting the claw. Let r_0 and r_1 denote the last $n-1$ bits of x_0 and x_1 respectively. The first bits satisfy $x_0[1] \oplus x_1[1] = s$ (the adaptive-hardcore property). Re-expressing the entangled state:

$$\begin{aligned} & \alpha_0 |0, x_0[1], r_0\rangle + \alpha_1 |1, x_1[1], r_1\rangle \\ &= \alpha_0 |0, x_0[1], r_0\rangle + \alpha_1 |1, x_0[1] \oplus s, r_1\rangle \\ &= \sum_a \alpha_a |a, x_0[1] \oplus as, r_a\rangle, \end{aligned}$$

where we have suppressed the index b for brevity; reinstating it gives

$$\sum_{a,b} \alpha_{a,b} |a, b, x_0[1] \oplus as, r_a\rangle.$$

CNOTing the third register onto the second produces

$$\sum_{a,b} \alpha_{a,b} |a, b \oplus x_0[1] \oplus as, x_0[1], r_a\rangle,$$

and noting that the third and fourth registers together carry x_a in full, we can write this as

$$(\mathbb{I} \otimes X^{x_0[1]} \otimes \mathbb{I}) \sum_{a,b} \alpha_{a,b} |a, b \oplus as, x_a\rangle.$$

Removing the claw via QFT. The state above contains the desired CNOT outcome $\sum_{a,b} \alpha_{a,b} |a, b \oplus as\rangle$ tensored with a residual $|x_a\rangle$. To unentangle the residual, apply a QFT (i.e. $H^{\otimes n}$) to the last n qubits:

$$\begin{aligned} & (\mathbb{I} \otimes X^{x_0[1]} \otimes H^{\otimes n}) (|0, b \oplus 0 \cdot s, x_0\rangle + |1, b \oplus 1 \cdot s, x_1\rangle) \\ &= \sum_d (-1)^{x_0 \cdot d} |0, b, d\rangle + (-1)^{x_1 \cdot d} |1, b \oplus s, d\rangle \\ &= \sum_d (-1)^{x_0 \cdot d} \left(|0, b, d\rangle + (-1)^{(x_0 \oplus x_1) \cdot d} |1, b \oplus s, d\rangle \right). \end{aligned}$$

Measure the last register to obtain a value d . The first two registers collapse to

$$|0, b\rangle + (-1)^{(x_0 \oplus x_1) \cdot d} |1, b \oplus s\rangle,$$

(up to a global phase). The desired CNOT outcome is $|a, b \oplus as\rangle$; we have it, modulo a Z correction whose exponent depends on $(x_0 \oplus x_1) \cdot d$.

Identifying the Pauli keys. The final state is

$$(Z^{(x_0 \oplus x_1) \cdot d} \otimes X^{x_0[1]}) \sum_{a,b} \alpha_{a,b} |a, b \oplus as\rangle.$$

This is precisely the desired CNOT outcome under a QOTP with classical Pauli exponents

$$z = (x_0 \oplus x_1) \cdot d, \quad x = x_0[1] \text{ (on the second qubit).}$$

The bit d is publicly known (it was measured). The Pauli keys depend on x_0 and x_1 , which only the holder of the TCF trapdoor can compute — but recall that we want *encryptions* of these keys, not the keys themselves. Since the classical HE scheme can homomorphically evaluate any classical circuit, and since FHE-encryptions of x_0 and x_1 are obtainable from the trapdoor (next subsection), we can compute $\text{Enc}(z)$ and $\text{Enc}(x)$ entirely within the FHE.

22.3 Implementing Step 1: From HE-Enc(s) to TCF-Enc(s)

It remains to produce the TCF pair (f_0, f_1) from an HE encryption of s .

Construction. Let Enc_{FHE} denote the homomorphic encryption scheme with public key pk and secret key sk . We define:

- $f_0(x_0) = \text{Enc}_{\text{FHE}}(x_0[1], r_{x_0})$, where $x_0[1]$ is the first bit and r_{x_0} is the rest of x_0 used as encryption randomness;
- $f_1(x_1) = f_0(x_1) \oplus \text{Enc}_{\text{FHE}}(s; r)$ for some randomness r .

The trapdoor is sk .

Verifying the hidden bit. Suppose $f_0(x_0) = f_1(x_1)$. Then

$$\text{Enc}_{\text{FHE}}(x_0[1]; r_{x_0}) = \text{Enc}_{\text{FHE}}(x_1[1]; r_{x_1}) \oplus \text{Enc}_{\text{FHE}}(s; r),$$

which by XOR-homomorphism rearranges to $x_0[1] = x_1[1] \oplus s$, i.e. $x_0[1] \oplus x_1[1] = s$. Hence the adaptive-hardcore property holds with hidden bit s .

Producing FHE-encryptions of the claw. Given y in the image, the trapdoor sk enables computing $x_0 = f_0^{-1}(y)$ and $x_1 = f_1^{-1}(y)$. Treating the inversion algorithm as a classical circuit and running it homomorphically on $\text{Enc}_{\text{FHE}}(sk)$ yields $\text{Enc}_{\text{FHE}}(x_0)$ and $\text{Enc}_{\text{FHE}}(x_1)$. From these and the publicly known d , the FHE scheme can homomorphically compute $\text{Enc}_{\text{FHE}}((x_0 \oplus x_1) \cdot d)$ and $\text{Enc}_{\text{FHE}}(x_0[1])$ — precisely the encrypted Pauli keys needed for the QOTP output. This completes the encrypted-CNOT construction.

22.4 Applications of Quantum FHE

Application 1: delegating BQP computations. A purely classical client wishes to outsource a quantum (BQP) computation to a quantum server while keeping its inputs private. The client publishes $\text{Enc}_{\text{FHE}}(N)$ to the server; the server runs Shor’s algorithm under the FHE, returning the classical FHE-encrypted output $\text{Enc}_{\text{FHE}}(p, q)$. The client decrypts to recover (p, q) , and the server never learns N .

Application 2: classical verification of quantum computation. A classical client wishes to delegate a quantum circuit C on input x to an untrusted quantum server, with C, x public, and to be assured the server actually performed the computation correctly. Returning the claimed output $y = C(x)$ is not enough; the client needs a method to verify it without running C itself. We will develop this protocol in the coming lectures.

Key Takeaways. The encrypted-CNOT operation is the central new ingredient required to handle the Toffoli gate in a quantum FHE scheme. It is implemented in two steps: (i) convert an FHE encryption of s into a TCF pair encrypting s via the adaptive-hardcore property, and (ii) entangle a claw with the quantum input, then apply a QFT and a measurement to leave the desired CNOT outcome under a known Pauli mask. The Pauli keys themselves are FHE-encrypted using the trapdoor, so the entire procedure leaves s hidden from the evaluator. The resulting quantum FHE scheme enables both blind delegation of BQP computations and (with further work) classical verification of quantum computation.

22.5 References

- Urmila Mahadev. *Classical Homomorphic Encryption for Quantum Circuits*. [arXiv:1708.02130](https://arxiv.org/abs/1708.02130).
- Lectures 20 and 21 from [OW15].

23 Testing A Qubit

This lecture begins our study of one of the foundational primitives in quantum cryptography from cryptographic assumptions: a classical-quantum protocol that “tests” whether a remote party is in possession of a genuine qubit. The technique is due to Brakerski, Christiano, Mahadev, Vazirani, and Vidick [BCM⁺18]; we follow the simplified treatment in [CCKW19] (CCKW).

23.1 Motivation



Figure 27: The overall setup: a classical client (left) interacts with a quantum server (right) to “test” a qubit on the server’s side.

Why test a qubit? Consider the scenario where Alice (the classical client) wants to interact with Bob (a quantum server) and use Bob’s quantum computer as a resource. She has no way to send quantum states (she has no quantum capability), but she can communicate classically and receive measurement outcomes. Without further constraints, Bob can pretend to perform quantum operations while actually running a classical simulation; for many problems this can be done efficiently when the input is a fixed classical string, so Alice has no way to certify that Bob is genuinely quantum.

The qubit-testing protocol will give Alice exactly that certification: a way to choose a measurement basis $\theta \in \{0, 1\}$ and force Bob to output a bit m that is approximately consistent with measuring a known qubit $|\psi\rangle$ in basis θ . Crucially, Bob must commit to (the measurement basis or the qubit) *before* learning θ , so he cannot adaptively choose to measure differently.

23.2 The Construction at a High Level

Suppose we have a TCF pair (f_0, f_1) with trapdoor td . Recall its properties:

- f_0, f_1 have the same image and are injective.
- No QPT can find a claw (x_0, x_1) with $f_0(x_0) = f_1(x_1)$ without td .
- Adaptive hardcore: for some hardcore predicate h , $h(x_0) \oplus h(x_1)$ is computationally hard to guess.

The protocol proceeds in four phases:

$$\begin{array}{ccc}
\mathcal{H} & \xrightarrow{V} & \mathbb{C}^2 \otimes \mathcal{H}' \\
\downarrow (X, Z) & & \downarrow (\sigma_x \otimes I, \sigma_z \otimes I) \\
\mathcal{H} & \xrightarrow{V} & \mathbb{C}^2 \otimes \mathcal{H}'
\end{array}$$

Figure 28: Isometry: the qubit Bob prepares is mapped to a quantum state that encodes a TCF preimage.

1. **Setup.** Alice samples a TCF pair (f_0, f_1) with trapdoor td and sends (f_0, f_1) to Bob.
2. **Commitment.** Bob prepares a superposition over $\{0, 1\}$ and inputs (similar to the claw-entangling step from Lecture 21), then applies U_{f_0, f_1} and measures the output register to obtain y . He sends y to Alice. At this point, his quantum register holds a superposition $\frac{1}{\sqrt{2}}(|0, x_0\rangle + |1, x_1\rangle)$ where (x_0, x_1) is the unique claw with $f_a(x_a) = y$.
3. **Challenge.** Alice samples a basis $\theta \xleftarrow{\$} \{0, 1\}$ and sends it to Bob.
4. **Response.** Bob measures his register in basis θ and returns the outcome.

Using the trapdoor, Alice can compute the claw (x_0, x_1) and verify that Bob's response matches a measurement of the qubit $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle$ in basis θ .

23.3 Detailed Protocol

Step 1: Bob prepares a uniform superposition. Bob initializes

$$\frac{1}{\sqrt{2|\mathcal{X}|}} \sum_{a \in \{0, 1\}} \sum_{x \in \mathcal{X}} |a, x\rangle.$$

Step 2: Apply U_{f_0, f_1} . Append a $|0\rangle$ ancilla and apply U_{f_a} which maps $|a, x, 0\rangle \rightarrow |a, x, f_a(x)\rangle$:

$$\frac{1}{\sqrt{2|\mathcal{X}|}} \sum_{a, x} |a, x, f_a(x)\rangle.$$

Step 3: Measure the third register. Bob measures the third register, obtaining some y in the common image. The first two registers collapse to

$$\frac{1}{\sqrt{2}}(|0, x_0\rangle + |1, x_1\rangle),$$

where $f_0(x_0) = f_1(x_1) = y$.

Step 4: Bob sends y to Alice. Since the TCF pair is claw-free (without td), Bob’s state is now genuinely entangled: he cannot “forget” which side of the claw he sampled.

Step 5: Alice sends the challenge basis θ .

- **Case $\theta = 0$ (computational basis).** Bob measures the second register, obtaining x_a for the corresponding a from the first register. Bob sends (a, x_a) to Alice. Alice computes $(x_0, x_1) = \text{Invert}(td, y)$ and verifies that the received x_a is one of them.
- **Case $\theta = 1$ (Hadamard basis).** Bob first applies H to all qubits in registers 1 and 2, then measures. We claim he obtains (a, d) where d is a random string satisfying $a \oplus d \cdot (x_0 \oplus x_1) = ?$ for some specified relation depending on the protocol. Bob sends (a, d) to Alice. Alice verifies the corresponding relation using her knowledge of x_0, x_1 .

Computing the Hadamard-basis outcome. The Hadamard of register 1 is straightforward; the Hadamard of register 2 (which encodes the claw bits) is the QFT over $\mathbb{F}_2^{|\mathcal{X}|}$, sending $|x_a\rangle \rightarrow \sum_d (-1)^{x_a \cdot d} |d\rangle$. Hence after Hadamards:

$$\begin{aligned} \frac{1}{\sqrt{2}} (H|0\rangle \otimes H|x_0\rangle + H|1\rangle \otimes H|x_1\rangle) &= \frac{1}{2\sqrt{|\mathcal{X}|}} \sum_d ((-1)^{x_0 \cdot d} |0, d\rangle + (-1)^{x_1 \cdot d} |1, d\rangle) \\ &\quad + \frac{1}{2\sqrt{|\mathcal{X}|}} \sum_d (-(-1)^{x_0 \cdot d} |1, d\rangle - (-1)^{x_1 \cdot d} |0, d\rangle). \end{aligned}$$

Conditional on measuring d , the first register collapses to a state $\propto |0\rangle + (-1)^{(x_0 \oplus x_1) \cdot d} |1\rangle$. The phase $\eta := (x_0 \oplus x_1) \cdot d$ thus characterizes the outcome. If $\eta = 0$, the post-measurement state is $|+\rangle$; if $\eta = 1$, it is $|-\rangle$.

Bob, who measures the first register in the Hadamard basis, then announces the corresponding outcome bit. Alice computes the same bit using (x_0, x_1) (which she can recover with td) and d (which Bob announced), and verifies consistency.

23.4 Why This Tests a Qubit

The protocol “tests a qubit” in the following sense: if Bob can answer Alice’s challenges correctly with high probability for *both* $\theta = 0$ and $\theta = 1$, he must be in possession of a quantum state that simultaneously encodes $|x_a\rangle$ outcomes (computational basis) and the parity $(x_0 \oplus x_1) \cdot d$ (Hadamard basis). A purely classical strategy cannot produce both: by the adaptive hardcore property, any classical Bob who knew enough about (x_0, x_1) to answer the Hadamard-basis challenge would have effectively found the claw.

More formally, the protocol implements a quantum analogue of *conjugate coding*, with the TCF pair playing the role of the basis choice and the trapdoor enabling Alice to verify.

23.5 Cryptographic Test of Quantumness

Iterating the qubit-testing protocol gives a *Test of Quantumness*: a classical protocol such that any quantum prover passes with high probability, but no classical prover passes with non-negligible probability. This is the central contribution of [BCM⁺18] and forms the basis for the classical verification of quantum computation, which we explore in future lectures.

Key Takeaways. A TCF pair allows a classical client to “encode” a qubit-equivalent challenge: the resulting protocol forces the server, with high probability, to behave consistently with measuring a qubit in either the computational or Hadamard basis. The trapdoor allows the client to verify outcomes she could otherwise not check. This primitive is the foundation for classical verification of quantum computation and certified randomness.

23.6 Further Reading

- Brakerski et al. [BCM⁺18].
- Cojocaru, Colisson, Kashefi, Wallden (CCKW) – a simplified treatment of [BCM⁺18], see [CCKW19].

24 Testing a Qubit – II; Circuit-to-Hamiltonian

In the previous lecture, we presented a protocol that “tests” whether a remote server is in possession of a genuine qubit. We sketched the structure but not the security argument. In this lecture, we first give that security argument: an adversarial server who passes the test must, except with negligible probability, hold a state close to a qubit. We then turn to the *circuit-to-Hamiltonian* reduction (also called Kitaev’s construction), the foundational technique for classical verification of arbitrary quantum computations.

24.1 Recap and Security Statement

Recall the qubit-testing protocol:

1. Alice samples a TCF pair (f_0, f_1) with trapdoor td and sends (f_0, f_1) to Bob.
2. Bob prepares a state, applies U_{f_0, f_1} , measures the image register to obtain y , and sends y to Alice. His remaining state is (in the honest case) $\frac{1}{\sqrt{2}}(|0, x_0\rangle + |1, x_1\rangle)$ for the claw $f_0(x_0) = f_1(x_1) = y$.
3. Alice samples $\theta \xleftarrow{\$} \{0, 1\}$ and sends to Bob.
4. Bob measures his register either in the computational basis ($\theta = 0$) or Hadamard basis ($\theta = 1$) and sends the outcome.
5. Alice uses td to verify consistency.

Theorem 24.1. *Suppose Bob passes the test with probability $1 - \varepsilon$. Then, conditioned on Alice’s θ choice and the outcome y , Bob’s residual state after committing is $O(\sqrt{\varepsilon})$ -close in trace distance to a state of the form $\frac{1}{\sqrt{2}}(|0, x_0\rangle + |1, x_1\rangle)$ (the honest state), for some isometry on Bob’s side mapping the qubit-of-interest into his Hilbert space.*

The theorem says: an adversarial Bob who passes the test almost always must have prepared a state essentially indistinguishable from the honest state. We sketch the argument below; full details are in [BCM⁺18] and [CCKW19].

24.2 Security Argument

Setup. Without loss of generality, after step 2 Bob holds a pure state $|\Psi\rangle_B$ on his side, together with the commitment y . Let $A_y |\Psi\rangle_B$ be Bob’s renormalized state conditioned on outputting y ; for clarity we suppress this conditioning notation.

Step 1: $\theta = 0$ check. On $\theta = 0$, Bob measures the appropriate register in the computational basis and outputs (a, x) . The check is: $f_a(x) = y$. Honest Bob produces (a, x_a) with a uniform; equality requires Bob's state to be supported on $\{|0, x_0\rangle, |1, x_1\rangle\}$ (the two preimages).

If Bob passes the $\theta = 0$ check with probability $1 - \varepsilon_0$, then by a standard ‘‘Gentle Measurement’’ argument, his state is $O(\sqrt{\varepsilon_0})$ -close to one supported entirely on this two-dimensional subspace. Hence we may write

$$|\Psi\rangle_B \approx \alpha |0, x_0\rangle \otimes |\phi_0\rangle_E + \beta |1, x_1\rangle \otimes |\phi_1\rangle_E,$$

where $|\phi_0\rangle_E, |\phi_1\rangle_E$ are arbitrary side states in some larger Hilbert space E that Bob also holds.

Step 2: $\theta = 1$ check. On $\theta = 1$, Bob applies H to both registers and measures, sending (a, d) . The check is: a equals the predicted parity $\eta = (x_0 \oplus x_1) \cdot d$ (recall this is how the Hadamard-basis outcome on the first register reveals the claw parity). Computing,

$$H^{\otimes(1+|x|)}(\alpha |0, x_0\rangle |\phi_0\rangle + \beta |1, x_1\rangle |\phi_1\rangle) = \sum_d \frac{1}{\sqrt{2^{|x|}}} (\alpha (-1)^{x_0 \cdot d} |+, d\rangle |\phi_0\rangle + \beta (-1)^{x_1 \cdot d} |-, d\rangle |\phi_1\rangle).$$

Measuring the second register yields a uniformly random d , and the first register collapses to

$$\propto \alpha (-1)^{x_0 \cdot d} |+\rangle |\phi_0\rangle + \beta (-1)^{x_1 \cdot d} |-\rangle |\phi_1\rangle.$$

The probability that Bob measures a matching the prediction η , summed over d , depends on the alignment between $|\phi_0\rangle$ and $|\phi_1\rangle$. If they are equal (i.e. Bob's side states are not entangled with the qubit register), the alignment is perfect and Bob passes with certainty. If they are orthogonal, Bob fails with probability $\frac{1}{2}$. By a careful argument (see [BCM⁺18]), passing with probability $1 - \varepsilon_1$ implies $|\langle \phi_0 | \phi_1 \rangle|^2 \geq 1 - O(\varepsilon_1)$, i.e. $|\phi_0\rangle$ and $|\phi_1\rangle$ are nearly identical.

Conclusion. Combining the two checks: $|\Psi\rangle_B \approx (\alpha |0, x_0\rangle + \beta |1, x_1\rangle) \otimes |\phi\rangle$ for some side state $|\phi\rangle$. The qubit register is unentangled with the rest of Bob's system and matches the honest construction. We have isolated a genuine qubit on Bob's side, up to an isometry that maps it into the appropriate computational sub-register.

24.3 Circuit-to-Hamiltonian Reduction

We now turn from testing qubits to verifying quantum computations. The first ingredient is Kitaev's *circuit-to-Hamiltonian* reduction.

24.3.1 Setup

Given a quantum circuit C on n input qubits, with T gates and producing a single-qubit output, we want to verify the claim that, on a known input $|x\rangle$, the output of C is (say) $|0\rangle$ with probability close to 1.

History state. Define the *history state* associated with C and $|x\rangle$ as

$$|\Phi\rangle = \frac{1}{\sqrt{T+1}} \sum_{t=0}^T |t\rangle_{\text{time}} \otimes |x_t\rangle_{\text{data}},$$

where $|x_0\rangle = |x\rangle$ is the input and $|x_t\rangle = U_t|x_{t-1}\rangle$ is the state after the t -th gate.

The history state has the key property that for any unitary U_t , $|t-1, x_{t-1}\rangle$ and $|t, x_t\rangle$ are related by U_t on the data register and an increment on the time register.

24.3.2 The Hamiltonian

Kitaev defines a 5-local Hamiltonian $H = H_{\text{in}} + H_{\text{prop}} + H_{\text{out}} + H_{\text{clock}}$ such that:

- H_{in} penalizes states whose data register at time 0 is not $|x\rangle$.
- H_{prop} penalizes states whose data register transitions do not match the circuit gates.
- H_{out} penalizes states whose data register at time T does not produce the claimed output.
- H_{clock} enforces the time register to be a unary counter.

The history state $|\Phi\rangle$ corresponding to the correct execution is the ground state of H with eigenvalue 0. Furthermore, if no quantum state has energy below some threshold (a *spectral gap*), then the circuit must not output the claimed value.

24.3.3 Why This Helps

The key insight is that verifying “does H have a ground state with energy below some threshold?” reduces to estimating an inner product, which (with the qubit-testing primitive) can be done by interacting classically with a quantum prover. The prover sends Alice a state that is supposed to be the history state; Alice tests it by measuring randomly chosen 5-local Hamiltonian terms, which can be done by selecting a few qubits and measuring them in the basis dictated by that term.

This high-level recipe — combined with the qubit-testing protocol so Alice can certify that the prover actually prepared the claimed state — yields a classical verification of arbitrary quantum (BQP) computations, due to Mahadev [Mah20].

Key Takeaways. The qubit-testing protocol of [BCM⁺18, CCKW19] comes with a rigorous security guarantee: any adversary who passes the test must hold a state nearly indistinguishable from the honest one, via an isometry from his Hilbert space. Combining this primitive with Kitaev’s circuit-to-Hamiltonian reduction lets a classical verifier check arbitrary quantum computations by inspecting random local terms of a Hamiltonian, with the prover certified to actually possess a genuine quantum state on which to evaluate them. This idea, originally due to Mahadev, is the foundation of classical verification of quantum computation.

24.4 Further Reading

- Brakerski et al. [BCM⁺18]: original cryptographic test of quantumness.
- Kitaev, Shen, Vyalyi: *Classical and Quantum Computation*, AMS 2002, Chapter 14 (5-local Hamiltonian).
- Mahadev [Mah20]: classical verification of quantum computation.

25 Classical Verification of Quantum Computation

We continue from last lecture toward the goal of classically verifying a quantum computation. We left off discussing the local Hamiltonian; before resuming, we identify the class of quantum computations we hope to verify, namely QMA.

Definition 27 (QMA). A language \mathcal{L} is in QMA if there exists a polynomial-size quantum verifier circuit C such that:

- if $x \in \mathcal{L}$, there exists a quantum witness $|\psi\rangle$ with $\Pr[C(x, |\psi\rangle) = 1] \geq \frac{2}{3}$;
- if $x \notin \mathcal{L}$, then for every quantum state $|\psi\rangle$, $\Pr[C(x, |\psi\rangle) = 1] \leq \frac{1}{3}$.

Here $|C| = \text{poly}(|x|)$.

This is the quantum analogue of the classical class NP, defined by

$$\mathcal{L} = \{x : \exists w \text{ such that } C(x, w) = 1\},$$

again with $|C| = \text{poly}(|x|)$.

Informally, NP is the class of languages for which, given x , there is a classical certificate w allowing efficient checking of language membership. QMA is the analogous class in which the certificate $|\psi\rangle$ is a *quantum* state and the checking circuit C is a quantum circuit. Our overall goal is to verify QMA computations *classically*.

25.1 Local Hamiltonian

Definition 28 (Local Hamiltonian associated to a circuit). Given a quantum circuit C with T gates and input x , the associated local Hamiltonian is

$$H_C = - \sum_{i \neq j} \frac{J_{ij}}{2} (\sigma_{x_i} \sigma_{x_j} + \sigma_{z_i} \sigma_{z_j}),$$

where the indices i, j range over $\text{poly}(T)$ qubits and J_{ij} are real coefficients efficiently computable from C and x .

H_C is a Hermitian operator that represents the “energy” of the circuit C on a given input.

Theorem 25.1 (Hamiltonian-energy characterization of acceptance). *There exist a value a and a gap $\delta \geq 1/\text{poly}(n)$ such that the following hold.*

- (I) *If there exists $|\psi\rangle$ with $\Pr[C(x, |\psi\rangle) = 1] \geq \frac{2}{3}$, then the smallest eigenvalue of H_C is at most a .*
- (II) *If for every $|\psi\rangle$, $\Pr[C(x, |\psi\rangle) = 1] \leq \frac{1}{3}$, then the smallest eigenvalue of H_C is at least $a + \delta$.*

The gap between cases (I) and (II) corresponds exactly to the gap between QMA completeness ($\geq 2/3$) and soundness ($\leq 1/3$). Standard amplification can widen this gap as needed; we henceforth assume each instance falls into case (I) or case (II).

Suppose the Prover wants to convince the Verifier that C outputs 1 with high probability on input x . The Prover therefore wants to demonstrate case (I): the smallest eigenvalue of H_C is at most a . If C does *not* accept x with high probability, we are in case (II), and the smallest eigenvalue is at least $a + \delta$. Our protocol must prevent the Prover from convincing the Verifier of case (I) when we are actually in case (II).

To check that the smallest eigenvalue of H_C is at most a , it suffices to check that there exists a quantum state $|\psi\rangle$ —namely the eigenstate of H_C with smallest eigenvalue—that passes certain statistical tests.

25.2 The Fitzsimons–Morimae Protocol

For now (ultimately we want a fully classical Verifier), assume the Verifier has the following restricted quantum capabilities:

- Hold one qubit in memory.
- Measure that qubit in the computational or Hadamard basis.

Both parties know (C, x) . The Prover claims to know $|\psi\rangle$ with $\Pr[C(x, |\psi\rangle) = 1] \geq \frac{2}{3}$ and is trying to convince the Verifier of this fact.

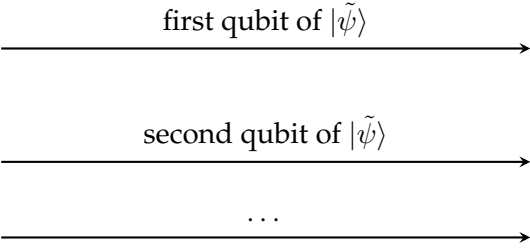
Prover

compute $H_C, \{J_{ij}\}_{\forall i,j}$

compute $|\tilde{\psi}\rangle$, the eigenstate
of H_C with smallest eigenvalue

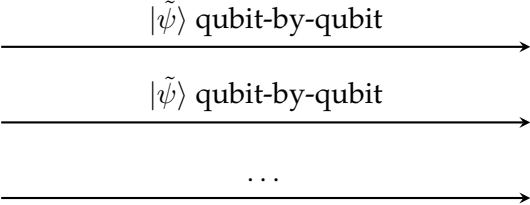
Verifier

compute $H_C, \{J_{ij}\}_{\forall i,j}$



- for each received qubit:
- measure in basis $W \xleftarrow{\$} \{X, Z\}$
 - save outcome

repeat the protocol many times:



repeat random measurements

then run statistical test using outcomes, J_{ij}, a
that PASSES iff
the eigenvalue of $|\tilde{\psi}\rangle$ is $\leq a$

This works because measuring each received qubit in a random basis from $\{X, Z\}$ corresponds to estimating the energy $\langle \tilde{\psi} | H_C | \tilde{\psi} \rangle$: H_C is a weighted sum of two-qubit Pauli operators, each of which is measured exactly by measuring the relevant qubits in the computational or Hadamard basis. The statistical test averages the outcomes (across all copies of $|\tilde{\psi}\rangle$ sent by the Prover), weights them by the J_{ij} , and checks whether the resulting estimate is at most a .

Note that picking $|\tilde{\psi}\rangle$ to be the eigenstate corresponding to the smallest eigenvalue of H_C is the Prover's optimal strategy: the Verifier accepts only when the estimated eigenvalue is at most a , so the Prover wants to minimize the eigenvalue of the state it sends.

Finally, to make this protocol work with a fully classical Verifier, two additional steps are

needed:

- Instead of having the Prover send qubits that the Verifier measures, the measurement task is delegated to the Prover. Using last lecture's qubit-testing protocol, the Verifier can be confident that the outcomes it receives are genuinely measurements of the desired qubit in the requested basis.
- The qubit-testing protocol from last lecture handles only a single qubit, so it must be extended to n qubits. This extension is the main technical step in the full Mahadev classical-verification protocol.

Key Takeaways.

- Verifying a QMA computation reduces, via Kitaev's circuit-to-Hamiltonian construction, to estimating the ground-state energy of a local Hamiltonian H_C .
- In the Fitzsimons–Morimae protocol, a Verifier with only single-qubit measurement capability can statistically estimate this energy from the Prover's claimed witness.
- Replacing the single-qubit Verifier with a fully classical one requires delegating the measurements to the Prover and certifying them via a qubit-testing protocol extended to n qubits.

25.3 Further Reading

- Lecture notes by Thomas Vidick, Lectures 5–7 [Vid20].

25.4 Quantum Money

We now turn to a related but distinct use of quantum mechanics in cryptography. As before, we consider the BB84 set of single-qubit states:

$$\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}.$$

Given an unknown state sampled from this set, it is impossible to copy by the no-cloning theorem [WZ82]. Wiesner proposed using this property of quantum states to construct banknotes that cannot be counterfeited.

Security goals.

1. Given a banknote, an adversary cannot produce a copy of it.
2. An adversary cannot generate a fresh banknote that the bank would accept as legitimate.

25.5 Model

Bank.

- Holds a secret key k .
- Issues authentic banknotes.

Banknote. A banknote consists of:

- A classical serial number S .
- A quantum state $|\psi_{f_k(S)}\rangle$ on n qubits (where n is the security parameter).

Each qubit of $|\psi\rangle$ is one of four states, indexed by two classical bits:

$$\begin{cases} |\psi_{00}\rangle = |0\rangle, \\ |\psi_{01}\rangle = |1\rangle, \\ |\psi_{10}\rangle = |+\rangle, \\ |\psi_{11}\rangle = |-\rangle. \end{cases}$$

The bank generates the quantum part of a banknote as follows:

1. Compute $y = f_k(S)$, where f_k is a keyed pseudorandom function with output length $2n$ bits.
2. Set the i th qubit of $|\psi\rangle$ to $|\psi_{y_{2i-1}, y_{2i}}\rangle$:
 - (a) the first qubit is $|\psi_{y_1 y_2}\rangle$;
 - (b) the second qubit is $|\psi_{y_3 y_4}\rangle$;
 - (c) and so on.

25.6 Verification

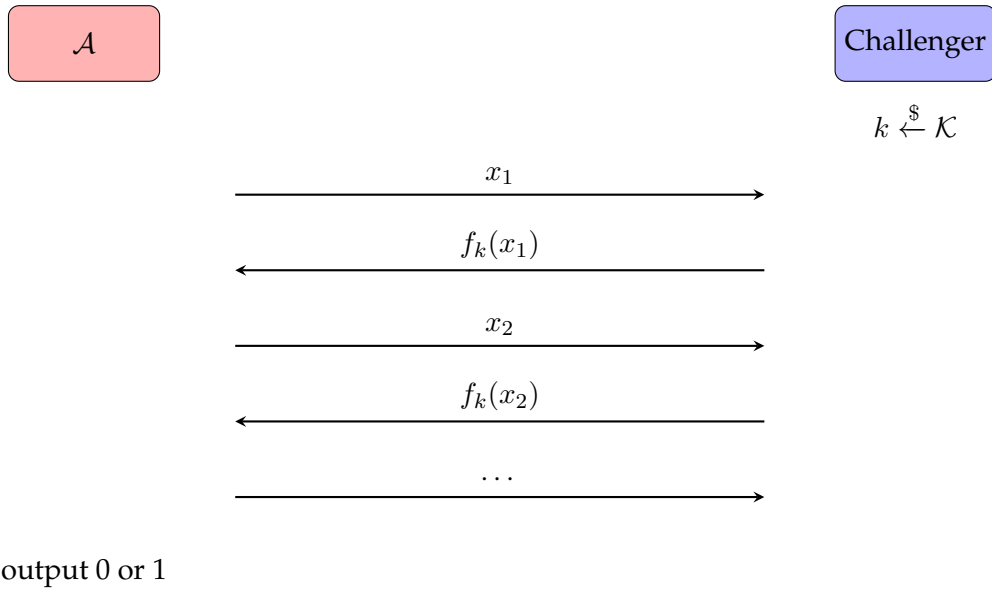
Given an alleged banknote $(S, |\psi\rangle)$, the bank verifies authenticity as follows:

1. Compute $y = f_k(S)$. Only the bank, holding k , can do this.
2. Check, qubit by qubit, that the i th qubit is $|\psi_{y_{2i-1}, y_{2i}}\rangle$.
 - Since the bank knows the basis (computational or Hadamard) of each qubit, it can measure each qubit in the “correct” basis, which does not disturb a legitimate state.
 - **Example.** Suppose the expected first qubit (per y) is $|0\rangle$, so the bank measures in the computational basis. If the actual first qubit is $|1\rangle$, the bank rejects with probability 1. If the actual first qubit is $|+\rangle$ or $|-\rangle$, the bank obtains outcome 0 (and proceeds) with probability $1/2$. Across n qubits, the probability that a banknote not equal to $|\psi_{f_k(S)}\rangle$ passes verification is $\text{negl}(n)$.

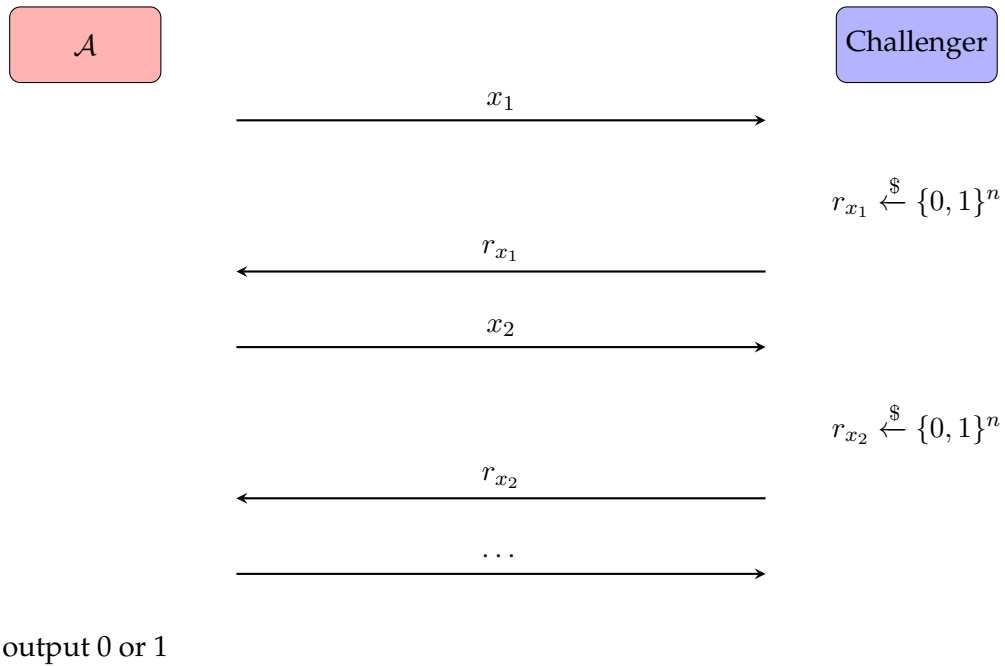
25.7 f_k is a Pseudorandom Function (PRF)

Security relies on f_k being a pseudorandom function (PRF). A keyed function family $\{f_k\}_{k \in \mathcal{K}}$ is a PRF if no efficient (quantum polynomial-size) adversary can distinguish the following two experiments with non-negligible probability:

Experiment 1 (real PRF)



Experiment 2 (random function)



In Experiment 2, the notation r_{x_1} indicates that if \mathcal{A} later repeats query x_1 , the challenger returns the same response without re-sampling.

A function family f is **pseudorandom** if, for every quantum polynomial-size adversary \mathcal{A} ,

$$|\Pr[\text{output}(\text{Experiment 1}) = 1] - \Pr[\text{output}(\text{Experiment 2}) = 1]| \leq \text{negl}(n).$$

We could in principle use a truly random function (as in Experiment 2), but storing one would require an exponentially large table. A pseudorandom function has a short description (the key k) and is indistinguishable from a truly random function, which suffices for our needs.

25.8 Security Intuition

To clone a banknote $(S, |\psi\rangle)$, an adversary \mathcal{A} would need to learn the state of each qubit of $|\psi\rangle$. Since they do not know the bank's key k , they cannot compute $y = f_k(S)$ (by PRF security, y is indistinguishable from a uniformly random string).

One natural attempt: just measure each qubit. But since the adversary does not know which basis each qubit lives in, they have to guess. For example, if the qubit is $|0\rangle$ and the adversary measures in the computational basis, they correctly recover 0. If the adversary measures in the Hadamard basis instead, they obtain $|+\rangle$ or $|-\rangle$ each with probability $\frac{1}{2}$, both of which are wrong (and the qubit is irreversibly disturbed).

Since the basis of each qubit is derived from the pseudorandom string y , the probability that the adversary guesses the correct basis for any particular qubit is $\frac{1}{2}$. Therefore, for an n -qubit state $|\psi\rangle$, the probability that the adversary correctly learns every qubit is at most $(\frac{1}{2})^n = \text{negl}(n)$.

Key Takeaways.

- Wiesner's quantum money exploits no-cloning: each banknote is a tensor product of BB84 states determined by a PRF applied to the serial number.
- Verification requires the bank's secret key, since only the bank knows which basis each qubit lives in. This is a *private-key* (or "private-verification") quantum money scheme.
- Security against cloning reduces to the fact that an adversary cannot guess the random bases for all n qubits except with probability 2^{-n} . (The next lecture analyzes a stronger adversary that interacts with the bank and gives a more nuanced security picture.)

26 Quantum Money (Continued); the Zeno Effect; Quantum Lightning; Cryptography with Deletion

26.1 Recap from Last Lecture

In the previous lecture, we introduced Wiesner's quantum money scheme, which constructs banknotes as quantum states. The scheme is motivated by the no-cloning theorem: a general unknown quantum state cannot be copied. The construction uses the familiar BB84 states, where each banknote is a quantum state $|\psi\rangle$ whose qubits are sampled from $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$ according to a secret string derived from the bank's PRF applied to the serial number s . Verification goes through the bank, which measures each qubit in the correct basis (known only to the bank).

We considered several variants of the scheme, differing mainly in how the bank handles a measurement mismatch. One particular variant—the *Wiesner strict-testing* or *good-money-returning, bad-money-confiscating* variant—confiscates the banknote whenever any qubit measurement disagrees with the expected value, alerts law enforcement, and apprehends the holder; otherwise, the bank returns the banknote to the customer. In this lecture, we gradually build up an attack on this variant.

26.2 The Quantum Zeno Effect

We first introduce an interesting phenomenon, known as the *quantum Zeno effect*. In subsequent sections we leverage this to solve the Elitzur–Vaidman bomb tester problem, which in turn inspires an attack on Wiesner's quantum money scheme.

Consider the problem of turning $|0\rangle$ into $|1\rangle$. If unitaries are available the task is trivial: applying the Pauli X (bit-flip) gate maps $|0\rangle \rightarrow |1\rangle$. What if we are restricted to using only measurements?

As a first step, consider what happens when we measure $|0\rangle$ in an orthogonal basis rotated counter-clockwise from the computational basis by a small angle ε (see Figure 29).

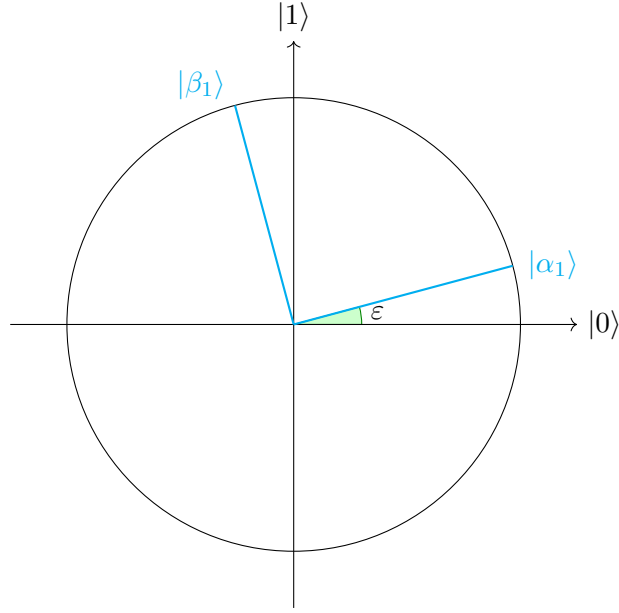


Figure 29: The basis $\mathcal{B}_1 = \{|\alpha_1\rangle, |\beta_1\rangle\}$ is obtained from the computational basis by a counter-clockwise rotation by an angle ε .

Formally, consider the orthonormal basis $\mathcal{B}_1 := \{|\alpha_1\rangle, |\beta_1\rangle\}$, where

$$|\alpha_1\rangle = \cos \varepsilon |0\rangle + \sin \varepsilon |1\rangle$$

and

$$|\beta_1\rangle = -\sin \varepsilon |0\rangle + \cos \varepsilon |1\rangle.$$

Let us verify orthonormality, i.e. $\langle \alpha_1 | \beta_1 \rangle = 0$ and $\langle \alpha_1 | \alpha_1 \rangle = \langle \beta_1 | \beta_1 \rangle = 1$:

$$\begin{aligned} \langle \alpha_1 | \beta_1 \rangle &= (\cos \varepsilon \langle 0| + \sin \varepsilon \langle 1|)(-\sin \varepsilon |0\rangle + \cos \varepsilon |1\rangle) \\ &= -\cos \varepsilon \sin \varepsilon \langle 0|0\rangle + \cos^2 \varepsilon \langle 0|1\rangle - \sin^2 \varepsilon \langle 1|0\rangle + \sin \varepsilon \cos \varepsilon \langle 1|1\rangle \\ &= 0, \end{aligned}$$

and

$$\begin{aligned} \langle \alpha_1 | \alpha_1 \rangle &= (\cos \varepsilon \langle 0| + \sin \varepsilon \langle 1|)(\cos \varepsilon |0\rangle + \sin \varepsilon |1\rangle) \\ &= \cos^2 \varepsilon \langle 0|0\rangle + \cos \varepsilon \sin \varepsilon \langle 0|1\rangle + \sin \varepsilon \cos \varepsilon \langle 1|0\rangle + \sin^2 \varepsilon \langle 1|1\rangle \\ &= \cos^2 \varepsilon + \sin^2 \varepsilon = 1. \end{aligned}$$

The calculation $\langle \beta_1 | \beta_1 \rangle = 1$ is similar and omitted.

When we measure $|0\rangle$ in \mathcal{B}_1 , the probability of outcome $|\alpha_1\rangle$ is (see [NC16], equation (2.103))

$$\begin{aligned} \Pr[\text{outcome is } |\alpha_1\rangle] &= \langle 0 | \alpha_1 \rangle \langle \alpha_1 | 0 \rangle \\ &= \langle 0 | (\cos \varepsilon |0\rangle + \sin \varepsilon |1\rangle)(\cos \varepsilon \langle 0| + \sin \varepsilon \langle 1|) |0\rangle \\ &= \cos^2 \varepsilon, \end{aligned}$$

and so the probability of outcome $|\beta_1\rangle$ is $1 - \cos^2 \varepsilon = \sin^2 \varepsilon$. Since $\lim_{\varepsilon \rightarrow 0} (\sin \varepsilon)/\varepsilon = 1$, for small ε we have $\sin \varepsilon \approx \varepsilon$. Therefore, with probability $\approx 1 - \varepsilon^2$ we observe $|\alpha_1\rangle$ (the “good” event), and with probability $\approx \varepsilon^2$ we observe $|\beta_1\rangle$ (the “bad” event). With high probability the good event occurs.

Why is the good event good? Because when it occurs, the measurement collapses $|0\rangle$ into $|\alpha_1\rangle = \cos \varepsilon |0\rangle + \sin \varepsilon |1\rangle$. The new state has a small amplitude $\sin \varepsilon$ on $|1\rangle$: if we now measure $|\alpha_1\rangle$ in the computational basis, with probability $\sin^2 \varepsilon \approx \varepsilon^2$ we obtain $|1\rangle$. Although we are not all the way there, we have made small progress toward turning $|0\rangle$ into $|1\rangle$. Geometrically, we have rotated $|0\rangle$ counter-clockwise by angle ε .

Now suppose the good event has happened. Consider another basis $\mathcal{B}_2 = \{|\alpha_2\rangle, |\beta_2\rangle\}$, obtained by rotating \mathcal{B}_1 counter-clockwise by another angle ε , i.e.

$$|\alpha_2\rangle = \cos \varepsilon |\alpha_1\rangle + \sin \varepsilon |\beta_1\rangle$$

and

$$|\beta_2\rangle = -\sin \varepsilon |\alpha_1\rangle + \cos \varepsilon |\beta_1\rangle.$$

A similar calculation shows this is also orthonormal; measuring $|\alpha_1\rangle$ in \mathcal{B}_2 yields $|\alpha_2\rangle$ with probability $\cos^2 \varepsilon \approx 1 - \varepsilon^2$ and $|\beta_2\rangle$ with probability $\approx \varepsilon^2$. Again, when the good event happens—i.e. we obtain $|\alpha_2\rangle$ —we have made another small step. Geometrically, we have rotated the original $|0\rangle$ state counter-clockwise by angle 2ε .

This process generalizes. Let $\mathcal{B}_n = \{|\alpha_n\rangle, |\beta_n\rangle\}$ be the basis obtained by rotating the computational basis counter-clockwise by $n\varepsilon$. At step n , we measure $|\alpha_{n-1}\rangle$ in \mathcal{B}_n ; the good event (outcome $|\alpha_n\rangle$) has probability $1 - \sin^2 \varepsilon \approx 1 - \varepsilon^2$. The probability that all n steps succeed is at least

$$(1 - \varepsilon^2)^n \geq 1 - n\varepsilon^2,$$

using Bernoulli’s inequality $(1 - x)^t \geq 1 - tx$ for $x \in [0, 1]$. After n successful steps, the final state $|\alpha_n\rangle$ is at angle $n\varepsilon$ from $|0\rangle$. Choosing $n = \pi/(2\varepsilon)$, we have rotated $|0\rangle$ all the way to $|1\rangle$ with probability

$$\Pr[\text{successful rotation } |0\rangle \rightarrow |1\rangle] \geq 1 - \frac{\pi}{2\varepsilon} \varepsilon^2 = 1 - \frac{\pi\varepsilon}{2}.$$

Taking ε small enough (e.g. $\varepsilon = 1/(500\pi)$) makes this success probability close to 1 (e.g. ≥ 0.999).

The above discussion illustrates the *quantum Zeno effect*: measurements in slightly rotated bases do not disturb the state by much, and this can be exploited to transform one quantum state into another, provided each rotation is small. We use this idea in the next section to tackle the bomb tester problem.

26.3 The Elitzur–Vaidman Bomb Tester Problem

Suppose we work at an airport security checkpoint and want to determine whether a suspicious-looking suitcase conceals a bomb. Suppose further that the only way to learn this classically is to open the suitcase—and if we do so and there is a bomb, it detonates. There is no good classical solution: never opening means letting a possible bomb through, while opening risks mass casualties. What if we are allowed to interact with the suitcase quantumly?

Formally, suppose we have an oracle (the suitcase) holding a hidden qubit, either $|0\rangle$ (no bomb) or $|1\rangle$ (bomb). We may query this oracle with any superposition state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. If the

hidden qubit is $|0\rangle$, the oracle returns $|\psi\rangle$ unchanged (this is crucial). If the hidden qubit is $|1\rangle$, the oracle measures $|\psi\rangle$ in the computational basis; on outcome 0 it returns $|0\rangle$, on outcome 1 it returns “EXPLODE” and the game is over.

Can we determine the hidden qubit without ever detonating the bomb? A naïve strategy that always queries $|0\rangle$ or $|1\rangle$ is unsafe or uninformative: it corresponds classically to either never opening or always opening. Querying $|+\rangle$ exposes us to a $1/2$ explosion probability when the bomb is present. With a more elaborate strategy, however, we can determine the hidden qubit completely and safely with high probability.

The technique uses the Zeno effect from above. Define $\mathcal{B}_n = \{|\alpha_n\rangle, |\beta_n\rangle\}_{n=1}^\infty$ as before. For the first query, set $|\psi_1\rangle = |\alpha_1\rangle = \cos \varepsilon |0\rangle + \sin \varepsilon |1\rangle$. Intuitively this is a safe query: measured in the computational basis it yields $|0\rangle$ with high probability for small ε , so the bomb-detonation risk is small.

If the hidden qubit is $|0\rangle$, the oracle returns $|\alpha_1\rangle$ unchanged. If the hidden qubit is $|1\rangle$, the oracle returns $|0\rangle$ with probability $\approx 1 - \varepsilon^2$, and the game ends with probability ε^2 . Notice that, conditional on survival, the states returned in the two cases differ by an angle of ε —this is the gap we exploit.

Let $|\psi'_1\rangle$ be the returned state. For the next step, rotate $|\psi'_1\rangle$ counter-clockwise by ε using

$$R_\varepsilon = \begin{bmatrix} \cos \varepsilon & -\sin \varepsilon \\ \sin \varepsilon & \cos \varepsilon \end{bmatrix}.$$

Let $|\psi_2\rangle = R_\varepsilon |\psi'_1\rangle$. If the hidden qubit was $|0\rangle$, then $|\psi'_1\rangle = |\alpha_1\rangle$, so $|\psi_2\rangle = |\alpha_2\rangle$. If the hidden qubit was $|1\rangle$, then $|\psi'_1\rangle = |0\rangle$, so $|\psi_2\rangle = |\alpha_1\rangle$. Querying the oracle with $|\psi_2\rangle$ yields $|\psi'_2\rangle = |\alpha_2\rangle$ (no bomb) or $|\psi'_2\rangle = |0\rangle$ (bomb, with survival probability $1 - \varepsilon^2$). The two cases now differ by angle 2ε , and so on.

We repeat for n steps: each round rotates the query result by ε , increasing the angular separation between the two cases by ε per round. The probability of surviving n rounds is at least $1 - n\varepsilon^2$. Taking $n = \pi/(2\varepsilon)$, we survive with probability at least $1 - \pi\varepsilon/2$. If we survive, we measure the final state $|\psi'_n\rangle$ in the computational basis: in the bomb-free case it has been rotated to $|1\rangle$, and in the bomb case it remained at $|0\rangle$. Thus we have **completely** determined the hidden qubit, i.e. detected the bomb without detonating it, with arbitrarily high probability (as $\varepsilon \rightarrow 0$).

This is a fun puzzle in its own right, but it has a cryptographic punchline: essentially the same idea attacks Wiesner’s quantum money scheme.

26.4 An Adaptive Attack on Wiesner’s Quantum Money¹¹

Before describing the attack, we reformulate the bomb-tester problem in a more convenient way. Suppose we (the security staff) hold two registers, called the *probe* register and the *system* register. The suitcase is modelled by the operator

$$C_P = |0\rangle\langle 0| \otimes \mathbb{I} + |1\rangle\langle 1| \otimes P,$$

where $P = \mathbb{I}$ if the suitcase is bomb-free and $P = X$ (the Pauli X) otherwise. We interact with the suitcase by applying C_P to our two registers. When the suitcase has a bomb, C_P is the CNOT operator (probe = control, system = target). When the suitcase is bomb-free, $C_P = |0\rangle\langle 0| \otimes \mathbb{I} + |1\rangle\langle 1| \otimes \mathbb{I} = \mathbb{I} \otimes \mathbb{I}$ is the identity.

Under this reformulation, our testing strategy becomes:

¹¹The presentation here follows [NSBU16], the original paper proposing the attack.

The reformulated bomb tester's strategy

1. Prepare the system register in state $|0\rangle$.
2. Rotate the probe register counter-clockwise by ε using R_ε .
3. Interact with the suitcase, i.e. apply C_P to the two registers.
4. Measure the system register. If the outcome is $|1\rangle$, the bomb explodes and we lose. Otherwise continue to the next round (go to step 1) **with the probe register unchanged**.

Figure 30: The 4-step bomb-tester strategy.

Let us analyse the two cases. If the suitcase is bomb-free, applying $C_P = \mathbb{I} \otimes \mathbb{I}$ in step 3 does nothing and leaves the two registers in state $\cos \varepsilon |00\rangle + \sin \varepsilon |10\rangle$. In step 4 we always measure $|0\rangle$ on the system register, and continue to the next round with the probe register in state $\cos \varepsilon |0\rangle + \sin \varepsilon |1\rangle$. After n rounds, the probe register is $\cos n\varepsilon |0\rangle + \sin n\varepsilon |1\rangle$.

If the suitcase contains a bomb, $C_P = |0\rangle\langle 0| \otimes \mathbb{I} + |1\rangle\langle 1| \otimes X$ in step 3 produces the state

$$\cos \varepsilon |00\rangle + \sin \varepsilon |11\rangle.$$

Measuring the system register in step 4 yields $|1\rangle$ (explosion) with probability $\sin^2 \varepsilon \approx \varepsilon^2$. If no explosion, the probe register collapses to $|0\rangle$.

This strategy is equivalent to the one in the previous section: each round increases the angular separation of the probe state across the two cases by ε . After $n = \pi/(2\varepsilon)$ rounds, survival probability is at least $1 - \pi\varepsilon/2$. Conditional on survival, the probe register is in $|1\rangle$ (bomb-free) or $|0\rangle$ (bomb). Measuring the probe register in the computational basis tells us, with arbitrarily high probability, whether the suitcase contained a bomb.

Application to Wiesner money. Recall that in Wiesner's scheme each banknote qubit is in one of $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$; the bank holds the secret string and verifies by measuring each qubit in the recorded basis. Under the strict-testing variant, any mismatch results in confiscation and arrest, and the bank returns the qubit to the customer if and only if measurement is consistent. An adversary wants to learn the state of each banknote qubit without alerting the bank—directly analogous to determining the suitcase's contents without exploding the bomb.

The attack treats each banknote qubit independently. The adversary uses a probe register and treats the banknote qubit as the system register. Without loss of generality, consider a single-qubit banknote in one of $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$. The strategy is the 4-step procedure in Figure 30, except that we **always apply CNOT** (i.e. $|0\rangle\langle 0| \otimes \mathbb{I} + |1\rangle\langle 1| \otimes X$) regardless of the unknown qubit, then send the system register (banknote qubit) to the bank and hope.

We analyze the four cases for the banknote (system) qubit:

1. If the banknote qubit was $|0\rangle$, the state before CNOT is $\cos \varepsilon |00\rangle + \sin \varepsilon |10\rangle$, and after CNOT is $\cos \varepsilon |00\rangle + \sin \varepsilon |11\rangle$. When the bank measures the banknote qubit (in the computational basis, since it knows the qubit is $|0\rangle$), it gets $|1\rangle$ with probability $\sin^2 \varepsilon \approx \varepsilon^2$, raising the alarm. Otherwise, the bank's measurement collapses the probe register to $|0\rangle$ and returns the banknote qubit $|0\rangle$.

2. If the banknote qubit was $|1\rangle$, the state before CNOT is $\cos \varepsilon |01\rangle + \sin \varepsilon |11\rangle$, and after CNOT is $\cos \varepsilon |01\rangle + \sin \varepsilon |10\rangle$. The bank's computational-basis measurement yields $|0\rangle$ with probability $\sin^2 \varepsilon \approx \varepsilon^2$, raising the alarm. Otherwise, the probe register collapses to $|0\rangle$ and the bank returns $|1\rangle$.
3. If the banknote qubit was $|+\rangle$, the state before CNOT is $(\cos \varepsilon |0\rangle + \sin \varepsilon |1\rangle) \otimes |+\rangle$. Applying CNOT has no effect on either register:

$$\begin{aligned}
& (|0\rangle \langle 0| \otimes \mathbb{I} + |1\rangle \langle 1| \otimes X) \left[(\cos \varepsilon |0\rangle + \sin \varepsilon |1\rangle) \otimes \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \right) \right] \\
&= (|0\rangle \langle 0| \otimes \mathbb{I} + |1\rangle \langle 1| \otimes X) \left(\frac{\cos \varepsilon |00\rangle + \cos \varepsilon |01\rangle + \sin \varepsilon |10\rangle + \sin \varepsilon |11\rangle}{\sqrt{2}} \right) \\
&= \frac{\cos \varepsilon |00\rangle + \cos \varepsilon |01\rangle + \sin \varepsilon |11\rangle + \sin \varepsilon |10\rangle}{\sqrt{2}} \\
&= (\cos \varepsilon |0\rangle + \sin \varepsilon |1\rangle) \otimes \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \right).
\end{aligned}$$

The state is unchanged. The bank's Hadamard-basis measurement yields $|+\rangle$ with certainty and returns $|+\rangle$. The probe register is now rotated by ε relative to its starting position.

4. If the banknote qubit was $|-\rangle$, the state before CNOT is $(\cos \varepsilon |0\rangle + \sin \varepsilon |1\rangle) \otimes |-\rangle$. Applying CNOT introduces a relative phase on the probe register:

$$\begin{aligned}
& (|0\rangle \langle 0| \otimes \mathbb{I} + |1\rangle \langle 1| \otimes X) \left[(\cos \varepsilon |0\rangle + \sin \varepsilon |1\rangle) \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \right] \\
&= (|0\rangle \langle 0| \otimes \mathbb{I} + |1\rangle \langle 1| \otimes X) \left(\frac{\cos \varepsilon |00\rangle - \cos \varepsilon |01\rangle + \sin \varepsilon |10\rangle - \sin \varepsilon |11\rangle}{\sqrt{2}} \right) \\
&= \frac{\cos \varepsilon |00\rangle - \cos \varepsilon |01\rangle + \sin \varepsilon |11\rangle - \sin \varepsilon |10\rangle}{\sqrt{2}} \\
&= (\cos \varepsilon |0\rangle - \sin \varepsilon |1\rangle) \otimes \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right).
\end{aligned}$$

The probe register has a phase flip on $|1\rangle$. The bank's Hadamard-basis measurement returns $|-\rangle$ with certainty and the banknote qubit comes back as $|-\rangle$. The probe register is now rotated by $-\varepsilon$ (i.e. ε **clockwise**). Carrying out a second round, similar calculations show the probe register ends in $\cos 2\varepsilon |0\rangle + \sin 2\varepsilon |1\rangle$: the phase flip on $|1\rangle$ is re-introduced and cancels back to a positive sign. The pattern continues: in odd rounds the phase on $|1\rangle$ is negative, in even rounds positive.

Summary of the attack. Carrying out the strategy for n rounds: if the banknote qubit was in the computational basis, we survive each round with probability $\geq 1 - \varepsilon^2$ (so $\geq 1 - n\varepsilon^2$ overall), and each round leaves the probe register reset to $|0\rangle$. This case is analogous to the bomb case. If the banknote qubit was in the Hadamard basis, we never trigger an alarm, and the final probe register is $\cos n\varepsilon |0\rangle \pm \sin n\varepsilon |1\rangle$, where the sign depends on whether the qubit was $|+\rangle$ or $|-\rangle$ and on parity of n . This case is analogous to the no-bomb case.

Setting $n = \pi/(2\varepsilon)$, with probability $\geq 1 - \pi\varepsilon/2$ we have not been caught and the probe register is $|0\rangle$ (banknote qubit was computational basis) or $\pm|1\rangle$ (banknote qubit was Hadamard basis). Measuring the probe register in the computational basis tells us the banknote qubit’s basis—the phase sign does not affect the measurement outcome. Knowing the correct basis, we can now measure the banknote qubit in that basis to recover its true state. Thus we have completely broken the strict-testing Wiesner scheme with arbitrarily high probability.

This concludes our analysis of the bomb-tester attack on Wiesner’s quantum money. The interested reader is referred to [NSBU16], which extends the attack to even infinite (or unknown) lists of possible banknote-qubit states. We now move on to a brief introduction to some recent directions in quantum money and cryptography with deletion.

26.5 Quantum Lightning

The *quantum lightning* scheme, proposed in [Zha19], is a public-key quantum money scheme. So far our discussion has focused on private-key (bank-only) verification, where only the bank can verify a banknote because only it knows the per-qubit basis. This is undesirable in practice: ideally, Alice should be able to verify on the fly that Bob sent her an authentic banknote, without contacting the bank. An ideal quantum money scheme thus requires some form of public-key cryptography, so that the verification algorithm can be run publicly.

Quantum lightning generates random states (“bolts”), each accompanied by a serial number, such that no adversary, even one given a bolt $|\zeta\rangle$, can produce another bolt verifying to the same serial number. Formally, the scheme has two public algorithms, STORM and VERIFY: STORM samples a random bolt $|\zeta\rangle$; VERIFY, given $|\zeta\rangle$, either accepts (and outputs an associated serial number) or rejects. Every honestly sampled bolt is accepted, and any honestly sampled bolt consistently verifies to the same serial number. The security guarantee is that no efficient adversary \mathcal{A} , with access to both STORM and VERIFY, can produce two distinct bolts $|\zeta_0\rangle \neq |\zeta_1\rangle$ that verify to the same serial number.

To use this as quantum money, each bolt produced by STORM is a banknote. The security guarantee says no one can forge money unless they sample a fresh bolt via STORM (which has a different serial number). Since STORM and VERIFY are public, Alice can run VERIFY on the banknote Bob sends her, detecting counterfeits without involving the bank.

26.6 Cryptography with Deletion

Cryptography with deletion asks: can we design an encryption scheme in which an adversary holding a ciphertext ct can be *provably* asked to delete the underlying plaintext? In classical public-key encryption, the ciphertext carries the plaintext’s information forever: given the secret key, the plaintext is uniquely recoverable from ct . So if an adversary stores ct now and later obtains the secret key—through computational advances (e.g. they gain unbounded computational power, the encryption scheme is broken)—they can retroactively recover the plaintext.

Can we instead require the adversary to delete the plaintext and prove they have deleted it by producing a *certificate of deletion*? If so, we could guarantee that no adversary, even with future unbounded computational power, can retroactively recover the plaintext.

A simple idea using quantum states: to encrypt a bit b , build a quantum state consisting of qubits in BB84 form. That is, the quantum part is $|\psi\rangle = |x\rangle_\theta$ where $x, \theta \in \{0, 1\}^n$ are binary strings:

the basis of the i th qubit is dictated by θ_i , and the basis vector by x_i . The full encryption of b has two parts—the quantum state $|\psi\rangle$ and the classical

$$classical := \text{Enc}\left(\theta, b \oplus \bigoplus_{i:\theta_i=0} x_i\right),$$

where Enc is a (classical) public-key encryption algorithm (we omit the public/secret keys for brevity). The big XOR is the parity of all bits encoded in computational-basis positions of $|\psi\rangle$, XORed with the secret bit. The full ciphertext is $\text{ct} = (|\psi\rangle, \text{classical})$.

To decrypt, apply Dec to classical to recover θ and the second component. Measure the qubits of $|\psi\rangle$ in the bases given by θ to recover the x_i 's. Then XOR the parity of the x_i at the $\theta_i = 0$ positions with the second component to recover b .

To request that an adversary delete the underlying plaintext, ask them to measure every qubit of $|\psi\rangle$ in the Hadamard basis and send us the outcomes. We check whether the Hadamard-basis outcomes at the $\theta_i = 1$ positions match our expectations, and reject any discrepancy. Intuitively, if the adversary cannot break the classical public-key encryption (at the moment we issue the request), they have no idea which positions of $|\psi\rangle$ were prepared in the Hadamard basis. To pass our check, they must genuinely measure many qubits in the Hadamard basis. But by an argument analogous to those used in QKD and QOT, this implies they have also measured many computational-basis qubits in the Hadamard basis, collapsing them to random outcomes and destroying the parity information needed to recover b . Hence the adversary cannot retroactively recover b from classical , even if they break the classical encryption later.

Key Takeaways.

- The quantum Zeno effect—small measurements barely disturbing a state—enables the Elitzur–Vaidman bomb tester to detect a bomb without detonating it, with success probability $\rightarrow 1$ as $\varepsilon \rightarrow 0$.
- Reformulated as a probe-system CNOT interaction, the same trick gives an adaptive attack on the strict-testing Wiesner scheme: the adversary learns the basis (and then the value) of each banknote qubit, breaking unforgeability with arbitrarily high probability.
- *Quantum lightning* ([Zha19]) goes beyond private-key Wiesner-style money to give a public-verification scheme: any user can verify a bolt without contacting the bank.
- *Cryptography with certified deletion* exploits no-cloning to allow Bob to convince Alice that he has irrecoverably destroyed a ciphertext—an after-the-fact privacy guarantee impossible classically.

References

- [ABB⁺14] Romain Alléaume, Cyril Branciard, Jan Bouda, Thierry Debuisschert, Mehrdad Dianati, Nicolas Gisin, Mark Godfrey, Philippe Grangier, Thomas Länger, Norbert Lütkenhaus, et al. Using quantum key distribution for cryptographic purposes: a survey. *Theoretical Computer Science*, 560:62–81, 2014.

- [BCM⁺18] Zvika Brakerski, Paul Christiano, Urmila Mahadev, Umesh Vazirani, and Thomas Vidick. A cryptographic test of quantumness and certifiable randomness from a single quantum device, 2018.
- [BF12] Niek J. Bouman and Serge Fehr. Sampling in a quantum population, and applications, 2012.
- [CCKW19] Alexandru Cojocaru, Lé o Colisson, Elham Kashefi, and Petros Wallden. QFactory: Classically-instructed remote secret qubits preparation. In *Lecture Notes in Computer Science*, pages 615–645. Springer International Publishing, 2019.
- [GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Annual Cryptology Conference*, pages 75–92. Springer, 2013.
- [Hal17] Shai Halevi. Homomorphic encryption. In *Tutorials on the Foundations of Cryptography*, pages 219–276. Springer, 2017.
- [ILL89] Russell Impagliazzo, Leonid Levin, and Michael Luby. Pseudo-random generation from one-way functions. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC ’89, page 12–24, New York, NY, USA, 1989. Association for Computing Machinery.
- [IR89] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 44–61, 1989.
- [KRS09] Robert König, Renato Renner, and Christian Schaffner. The operational meaning of min-and max-entropy. *IEEE Transactions on Information theory*, 55(9):4337–4347, 2009.
- [Mah20] Urmila Mahadev. Classical homomorphic encryption for quantum circuits. *SIAM Journal on Computing*, (0):FOCS18–189, 2020.
- [MS77] Florence Jessie MacWilliams and Neil James Alexander Sloane. *The Theory of Error Correcting Codes*, volume 16. Elsevier, 1977.
- [NC16] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information (10th Anniversary edition)*. Cambridge University Press, 2016.
- [NSBU16] Daniel Nagaj, Or Sattath, Aharon Brodutch, and Dominique Unruh. An adaptive attack on wiesner’s quantum money. *Quantum Info. Comput.*, 16(11–12):1048–1070, sep 2016.
- [OW15] Ryan O’Donnell and John Wright. Quantum computation lecture notes. 2015.
- [Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), sep 2009.
- [Ren05] Renato Renner. *Security of Quantum Key Distribution*. PhD thesis, ETH Zürich, Zürich, Switzerland, 2005.

- [Vid20] Thomas Vidick. Interactions with quantum devices. 2020.
- [WZ82] W. K. Wootters and W. H. Zurek. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, 1982.
- [Zha19] Mark Zhandry. Quantum lightning never strikes the same state twice. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019*, pages 408–438, Cham, 2019. Springer International Publishing.