Quantum Circuits and Universality

Lectures 5 and 6

In lectures 3 and 4 we built the mathematics: states are unit vectors, gates are unitary operators, and several systems combine by the tensor product. Lectures 7, 8 and 9 we will build algorithms. These two lectures are meant to be the bridge between the two. We will attempt to answer 3 major questions throughout.

First, we introduce the *circuit model*. This is the notation in which every later lecture is written. It comes with a few identities that let you rewrite one circuit as another.

Second, we answer the question that Lecture 4 left open. A quantum computer applies unitary operators, but *which* unitaries can it apply? The answer is all of them, from a very short list of gates. The proof is a direct application of the spectral theorem that you already have.

Third, we show how ordinary classical computation is carried out inside a quantum circuit. This is what supplies the oracles that Lecture 7 starts to query on its first page.

The emphasis will be on smaller circuits that can be reasonably drawn. We will also present everything as an equation in addition to the circuit formulation.

Lecture	Block	Contents
5	1	The circuit model, conventions, gates, the Pauli basis, two-qubit gates and the core circuit identities, Bell states, GHZ states, partial measurement
5	2	$U = \mathrm{e}^{\mathrm{i}H}$, rotation gates, global versus relative phase, the Clifford group and conjugation, controlled gates and how to build them, phase kickback
6	3	Pauli exponentials as circuits, Trotter–Suzuki, universality, other universal gate sets, Solovay–Kitaev, why universality is not efficiency
6	4	Reversible classical computation, the Boolean oracle, no-cloning, compute, copy and uncompute, the phase oracle, measurement in circuits, counting cost

Contents

Cheat Sheet **3**

I Lecture 5: The Circuit Model **6**

1 Block 1: Circuits, Gates, and the Pauli Basis **6**
1.1 What a Circuit Is . . . 7
1.2 The Single-Qubit Gate Zoo . . . 8
1.3 The Pauli Relations . . . 8
1.4 Everything Is a Sum of Paulis . . . 9
1.5 Two-Qubit Gates . . . 11
1.6 Bell States and Correlations . . . 12
1.7 More Than Two Qubits: GHZ States and Partial Measurement . . . 13

2 Block 2: Rotations, Cliffords, and Controlled Gates **15**
2.1 Every Unitary Is an Exponential . . . 15
2.2 Global and Relative Phase . . . 16
2.3 The Clifford Group . . . 17
2.4 Controlled Gates and How to Build Them . . . 18
2.5 Phase Kickback, the Two-Qubit Case . . . 19

II Lecture 6: Universality and Computation **21**

3 Block 3: Universality **21**
3.1 A Pauli Exponential Is a Circuit . . . 21
3.2 Why You Cannot Just Multiply Exponentials . . . 23
3.3 The Universality Theorem . . . 24
3.4 Other Universal Gate Sets . . . 25
3.5 Universality Is Not Efficiency . . . 25
3.6 Gottesman–Knill . . . 26

4 Block 4: Classical Computation Inside a Quantum Circuit **26**
4.1 Reversibility, and the Shape of U_f . . . 27
4.2 No-Cloning . . . 27
4.3 Garbage, and How to Deal With It . . . 28
4.4 The Phase Oracle . . . 29
4.5 Measurement in Circuits . . . 30
4.6 Counting Cost (What is Hidden by the Oracle) . . . 31
4.7 Superstitions about Superposition . . . 31

Cheat Sheet

Conventions used throughout these two lectures. Getting these wrong is the commonest source of error, so they are collected first.

Item	Convention
time	circuits read left to right
wire order	qubit 1 at the top and *most* significant. $j = j_1 \cdots j_n$ means $j = \sum_l j_l 2^{n-l}$. *Opposite to Qiskit*
circuit vs matrix	the circuit "A then B then C" is the operator CBA
wires	single = qubit, double = classical bit, $/^n$ = a bundle of n qubits
controls	filled dot fires on $\|1\rangle$, open dot on $\|0\rangle$ (= filled dot conjugated by X)
meter	measurement in the computational (Z) basis
rotations	$R_{\hat{n}}(\theta) = \mathrm{e}^{-\mathrm{i}\theta(\hat{n}\cdot\sigma)/2}$, *minus* in the exponent. It rotates the Bloch vector by $+\theta$
$U_f, U_f^{\pm}$	Boolean oracle $\|x\rangle\|y\rangle \mapsto \|x\rangle\|y \oplus f(x)\rangle$. phase oracle $\|x\rangle \mapsto (-1)^{f(x)}\|x\rangle$

The gate zoo.

Gate	Matrix	Relations
X, Y, Z	$\left[\begin{smallmatrix}0&1\\1&0\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}0&-\mathrm{i}\\\mathrm{i}&0\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}1&0\\0&-1\end{smallmatrix}\right]$	$\sigma_j^2 = I$, $\sigma_j^\dagger = \sigma_j$, eigenvalues ± 1
H	$\frac{1}{\sqrt{2}}\left[\begin{smallmatrix}1&1\\1&-1\end{smallmatrix}\right]$	$H = \frac{X+Z}{\sqrt{2}}$, $H^2 = I$, $H = \mathrm{i}R_{(\hat{x}+\hat{z})/\sqrt{2}}(\pi)$
$P(\varphi)$	$\left[\begin{smallmatrix}1&0\\0&\mathrm{e}^{\mathrm{i}\varphi}\end{smallmatrix}\right]$	$Z = P(\pi)$, $S = P(\frac{\pi}{2})$, $T = P(\frac{\pi}{4})$, $R_k = P(2\pi/2^k)$
$R_z(\theta)$	$\left[\begin{smallmatrix}\mathrm{e}^{-\mathrm{i}\theta/2}&0\\0&\mathrm{e}^{\mathrm{i}\theta/2}\end{smallmatrix}\right]$	$P(\varphi) = \mathrm{e}^{\mathrm{i}\varphi/2}R_z(\varphi)$, $R_x(2\pi) = -I$
CNOT	$\|0\rangle\langle 0\| \otimes I + \|1\rangle\langle 1\| \otimes X$	$\|x\rangle\|y\rangle \mapsto \|x\rangle\|y \oplus x\rangle$, $\mathrm{CNOT}^2 = I$
CZ	$\mathrm{diag}(1,1,1,-1)$	symmetric in its two qubits
TOFFOLI	$\Lambda^2(X)$	$\|a,b,c\rangle \mapsto \|a,b,c \oplus (a \wedge b)\rangle$. 6 CNOTs, 7 T gates

Pauli algebra.

$$XY = \mathrm{i}Z, \quad YZ = \mathrm{i}X, \quad ZX = \mathrm{i}Y \qquad \text{(reverse any product and the sign flips)} \qquad XYZ = \mathrm{i}\,I$$

$$M = c_0 I + c_1 X + c_2 Y + c_3 Z, \quad c_j = \tfrac{1}{2}\operatorname{tr}(\sigma_j M). \qquad M = \sum_P c_P P \ \text{ on } n \text{ qubits}, \ c_P = 2^{-n}\operatorname{tr}(PM)$$

$$|0\rangle\langle 0| = \tfrac{I+Z}{2}, \quad |1\rangle\langle 1| = \tfrac{I-Z}{2}, \quad |0\rangle\langle 1| = \tfrac{X+\mathrm{i}Y}{2}, \quad |1\rangle\langle 0| = \tfrac{X-\mathrm{i}Y}{2}$$

Two Pauli strings commute or anticommute. They anticommute exactly when the number of positions carrying *different* non-identity Paulis is odd.

Clifford conjugation ($P \mapsto UPU^\dagger$). Generated by $\{H, S, \mathrm{CNOT}\}$.

	$X \mapsto$	$Y \mapsto$	$Z \mapsto$
H	Z	$-Y$	X
S	Y	$-X$	Z
X	X	$-Y$	$-Z$
Z	$-X$	$-Y$	Z

CNOT: control c, target t	
$X_c \mapsto X_c X_t$	$X_t \mapsto X_t$
$Z_c \mapsto Z_c$	$Z_t \mapsto Z_c Z_t$
$Y_c \mapsto Y_c X_t$	$Y_t \mapsto Z_c Y_t$
X forwards, Z backwards	

Circuit identities.

Name	Statement
CZ from CNOT	$\mathrm{CZ} = (I \otimes H)\,\mathrm{CNOT}_{1\to 2}\,(I \otimes H)$
control reversal	$(H \otimes H)\,\mathrm{CNOT}_{1\to 2}\,(H \otimes H) = \mathrm{CNOT}_{2\to 1}$
three-CNOT swap	$\mathrm{SWAP} = \mathrm{CNOT}_{1\to 2}\mathrm{CNOT}_{2\to 1}\mathrm{CNOT}_{1\to 2}$
Toffoli and CCZ	$\mathrm{TOFFOLI} = (I \otimes I \otimes H)\,\mathrm{CCZ}\,(I \otimes I \otimes H)$
conjugation into the exponent	$V\mathrm{e}^{A}V^{\dagger} = \mathrm{e}^{VAV^{\dagger}}$
rotation trading	$R_z(\theta) = HR_x(\theta)H, \quad R_y(\theta) = SR_x(\theta)S^{\dagger}$
controlled phase	$\Lambda(P(\theta))$: two CNOTs and three $P(\theta/2)$, one inverted
controlled-U	$U = \mathrm{e}^{\mathrm{i}\alpha}AXBXC$ with $ABC = I$. Two CNOTs, three gates, one $P(\alpha)$
doubly controlled	$\Lambda^2(U)$ from $\Lambda(V), \mathrm{CNOT}, \Lambda(V^{\dagger}), \mathrm{CNOT}, \Lambda(V)$ with $V^2 = U$
Pauli exponential	$\mathrm{e}^{-\mathrm{i}\theta P/2}$: basis change, CNOT ladder, $R_z(\theta)$, undo. $2(w-1)$ CNOTs
errors add	$\|U_m \cdots U_1 - V_m \cdots V_1\| \le \sum_j \|U_j - V_j\|$

Entanglement from circuits.

Object	Statement
Bell circuit	H on qubit 1, then CNOT. It sends $\|x, y\rangle$ to the four Bell states
Bell states	$\|\Phi^{\pm}\rangle = (\|00\rangle \pm \|11\rangle)/\sqrt{2}, \quad \|\Psi^{\pm}\rangle = (\|01\rangle \pm \|10\rangle)/\sqrt{2}$
two bases	$\|\Phi^{+}\rangle = (\|{+}{+}\rangle + \|{-}{-}\rangle)/\sqrt{2}$: agrees in Z and in X
no product state	can agree with certainty in two different bases
product test	$\|\psi\rangle$ is a product state iff $a_{00}a_{11} = a_{01}a_{10}$
GHZ	$(\|0\cdots 0\rangle + \|1\cdots 1\rangle)/\sqrt{2}$. Fan-out or chain of CNOT gates
partial meas.	$P_k = \|k\rangle\langle k\| \otimes I \otimes \cdots$, $p(k) = \|P_k\,\|\psi\rangle\|^2$, state $P_k\,\|\psi\rangle\,/\sqrt{p(k)}$
GHZ, Z on one qubit	leaves $\|0\cdots 0\rangle$ or $\|1\cdots 1\rangle$: a product state
GHZ, X on one qubit	leaves $(\|0\cdots 0\rangle \pm \|1\cdots 1\rangle)/\sqrt{2}$: still a GHZ state

Universality.

Result	Statement
$U = \mathrm{e}^{\mathrm{i}H}$	every unitary is the exponential of a Hermitian, and conversely
Pauli basis	$H = \sum_j J_j P_j$ with J_j *real* iff H Hermitian
Trotter–Suzuki	$\left(\prod_j \mathrm{e}^{\mathrm{i}A_j/N}\right)^N \to \mathrm{e}^{\mathrm{i}\sum_j A_j}$, error $\mathcal{O}(1/N)$
obstruction	$\mathrm{e}^{A}\mathrm{e}^{B} = \mathrm{e}^{A+B} + \frac{1}{2}[A, B] + \cdots$
universal sets	$\{R_x(\theta), H, S, \mathrm{CNOT}\}$, Clifford + T, Toffoli + H, Clifford + any non-Clifford
Solovay–Kitaev	accuracy ϵ from a finite set costs $\mathcal{O}(\log^c(1/\epsilon))$ gates
counting bound	almost every n-qubit unitary needs $\Omega(4^n/\log n)$ gates
Gottesman–Knill	Clifford circuits on basis states with Z measurements are classically simulable

Classical computation inside a circuit.

Result	**Statement**
why $\oplus$	$\lvert x\rangle \mapsto \lvert f(x)\rangle$ is unitary only for bijective f. U_f is unitary for every f, and $U_f^2 = I$
Toffoli as logic	target $\lvert 0\rangle$ gives AND, target $\lvert 1\rangle$ gives NAND, CNOT with target $\lvert 0\rangle$ copies a bit
no-cloning	no unitary sends $\lvert\psi\rangle \lvert 0\rangle \mapsto \lvert\psi\rangle \lvert\psi\rangle$ for all $\lvert\psi\rangle$. The proof is $z = z^2$
uncomputation	compute, copy with CNOTs, run $U^\dagger$. $\mathcal{O}(g)$ gates, no garbage
garbage is fatal	distinct $g(x)$ makes every outcome equally likely: $p = 2^{-n}$ whatever f is
phase kickback	$U_f \lvert x\rangle \lvert -\rangle = (-1)^{f(x)} \lvert x\rangle \lvert -\rangle$
general kickback	$U \lvert u\rangle = \mathrm{e}^{2\pi \mathrm{i}\varphi} \lvert u\rangle \Rightarrow \Lambda(U) \lvert c\rangle \lvert u\rangle = \mathrm{e}^{2\pi \mathrm{i} c\varphi} \lvert c\rangle \lvert u\rangle$
Hadamard test	measuring the control in $\{\lvert\pm\rangle\}$ gives + with probability $\cos^2(\pi\varphi)$
deferred meas.	you can replace a measured control by a quantum control and move the measurement to the end
implicit meas.	you can assume an unmeasured wire at the end was measured, so garbage decoheres
other bases	to measure $\{\lvert v_j\rangle = V \lvert j\rangle\}$, apply $V^\dagger$ then measure. For X, apply H

Costs. Count gates over a stated set, and count depth. The CNOT count dominates on today's hardware, and the T-count dominates on an error-corrected machine. Width counts qubits including ancillas, and the query count charges nothing for building the oracle.

Part I

Lecture 5: The Circuit Model

In Lecture 4 we described the machine in the language of linear algebra. A state is a unit vector in $(\mathbb{C}^2)^{\otimes n}$. An operation is a unitary matrix of size $2^n \times 2^n$. A measurement is a set of projectors. This description is correct, but it is hard to use. Nobody can write down a $2^n \times 2^n$ matrix, and nobody can build one.

The circuit model solves this problem. It describes a large unitary by a recipe instead of by its entries. The recipe is a short list of operations, and each operation acts on one or two qubits. In this lecture we set up the language of circuits and fill it with gates. In the next lecture we prove that these recipes are enough.

1 Block 1: Circuits, Gates, and the Pauli Basis

By the end of this block you can. . .

- Read and draw a quantum circuit diagram. You can also state the conventions that we use for wire order and for the relation between circuit order and matrix order.
- Write down X, Y, Z, H, S, T and the phase gate $P(\varphi)$ as matrices and as actions on the computational basis.
- Use the Pauli multiplication and anticommutation relations.
- Expand any 2×2 matrix in the basis $\{I, X, Y, Z\}$. You can also expand any $2^n \times 2^n$ matrix in the basis of n-qubit Pauli strings.
- Draw and manipulate CNOT, CZ and SWAP, and prove the three identities that relate them.
- Build the Bell states with two gates, and say what makes their correlations different from anything a product state can do.
- Build the GHZ state on four qubits in two ways, and measure one qubit of a register while leaving the others alone.

Why a diagram and not a matrix

A general n-qubit unitary has 4^n real parameters. For $n = 10$ that is about a million numbers. For $n = 50$ it is more numbers than there are atoms in a person. Nobody writes an algorithm that way.

Instead, we write an algorithm as a *circuit*. A circuit is a list of a few hundred or a few million gates. Each gate touches one or two qubits, and the product of all of them is the unitary that we want.

The whole subject lives in the gap between these two descriptions. A unitary with a short circuit is cheap. A unitary without one cannot be built. In Section 3.5 we show that almost every unitary is of the second kind. So when we think in circuits, we think about which unitaries are cheap.

1.1 What a Circuit Is

A quantum circuit is a diagram. Each horizontal line is a *wire*, and each wire carries one qubit. The boxes on the wires are *gates*. You read the diagram from left to right. The left edge holds the input state.

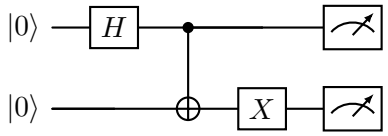


Figure 1: The parts of a circuit. Two wires mean two qubits. The labels on the left give the input state $|0\rangle \otimes |0\rangle$. Time runs to the right. The meter symbols are measurements in the computational basis.

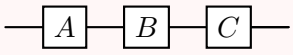
Six conventions make the diagrams unambiguous. They are not universal, and other sources use other conventions. The most common source of sign errors and ordering errors is a mix of two different conventions. We collect all six here, and we use them for the rest of the course without further comment.

Conventions.

1. Time runs from left to right.
2. The top wire is qubit 1, and qubit 1 is the *most* significant bit. So $|j_1 j_2 \cdots j_n\rangle$ has j_1 at the top, and $j = \sum_l j_l 2^{n-l}$. This is the opposite of the Qiskit convention. We use it in Lectures 7 to 9, so we adopt it now. Some sources number the wires from 0 instead of from 1. The order is the same, and only the label changes: their qubit 0 is our qubit 1.
3. Circuit order is the reverse of matrix order. The circuit "A, then B, then C" is the operator CBA.
4. A single line is a qubit. A doubled line is a classical bit. A line crossed by $/^n$ carries n qubits at once.
5. A filled control dot fires on $|1\rangle$. An open control dot fires on $|0\rangle$. An open dot is a filled dot conjugated by X.
6. The meter symbol is a measurement in the computational basis. Any other measurement is drawn as a basis change and then a meter.

The order reversal

Convention 3 catches everyone once. Take the circuit



The state that leaves on the right is $CBA\,|\psi\rangle$. The gate A acts first, and operators compose from right to left.

If you read a circuit aloud from left to right and then write the matrices in that order, you get the reversed product. For gates that do not commute, that is a different operator. When you check a circuit by hand, choose one method. Track the state through the diagram from left to right, or write the matrix product from right to left. Do not do both at once.

Remark 1.1 (A note on what circuits are)**.** A circuit describes a unitary. It is not necessarily a sequence of events. Nothing in the formalism says that the gates happen at different times. Gates on disjoint qubits commute, as we saw previously. So you can draw them in either order or stack them in one column. A circuit does record its *depth*, which is the number of columns of non-commuting gates. Depth is the right measure of running time, and we count it in Section 4.6.

1.2 The Single-Qubit Gate Zoo

The adjoint of a matrix, also called its Hermitian conjugate, is its transpose-conjugate. We defined it in Lecture 4:

$$M = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \Longrightarrow \quad M^{\mathrm{T}} = \begin{bmatrix} a & c \\ b & d \end{bmatrix}, \quad M^{*} = \begin{bmatrix} a^{*} & b^{*} \\ c^{*} & d^{*} \end{bmatrix}, \quad M^{\dagger} = (M^{*})^{\mathrm{T}} = \begin{bmatrix} a^{*} & c^{*} \\ b^{*} & d^{*} \end{bmatrix}.$$

A gate is a unitary matrix, so $U^{\dagger}U = UU^{\dagger} = I$. Six gates account for almost everything that we draw in this course.

Gate	Matrix	Action on the basis	Notes
X	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	$\lvert 0\rangle \mapsto \lvert 1\rangle$, $\lvert 1\rangle \mapsto \lvert 0\rangle$	the NOT gate
Y	$\begin{bmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{bmatrix}$	$\lvert 0\rangle \mapsto \mathrm{i}\lvert 1\rangle$, $\lvert 1\rangle \mapsto -\mathrm{i}\lvert 0\rangle$	$Y = \mathrm{i}XZ$
Z	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	$\lvert 0\rangle \mapsto \lvert 0\rangle$, $\lvert 1\rangle \mapsto -\lvert 1\rangle$	the phase flip
H	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	$\lvert 0\rangle \mapsto \lvert +\rangle$, $\lvert 1\rangle \mapsto \lvert -\rangle$	$H = \dfrac{X+Z}{\sqrt{2}}$, $H^2 = I$
S	$\begin{bmatrix} 1 & 0 \\ 0 & \mathrm{i} \end{bmatrix}$	$\lvert 1\rangle \mapsto \mathrm{i}\lvert 1\rangle$	$S^2 = Z$, so $S = \sqrt{Z}$
T	$\begin{bmatrix} 1 & 0 \\ 0 & \mathrm{e}^{\mathrm{i}\pi/4} \end{bmatrix}$	$\lvert 1\rangle \mapsto \mathrm{e}^{\mathrm{i}\pi/4}\lvert 1\rangle$	$T^2 = S$, so $T = \sqrt{S}$

The gates S and T are special cases of the *phase gate*, which multiplies $|1\rangle$ by a phase and leaves $|0\rangle$ alone:

$$P(\varphi) = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{e}^{\mathrm{i}\varphi} \end{bmatrix}, \qquad Z = P(\pi), \quad S = P(\pi/2), \quad T = P(\pi/4). \tag{1}$$

In Lecture 7 we need the family $R_k = P(2\pi/2^k)$, so that $R_1 = Z$, $R_2 = S$ and $R_3 = T$. It is the same gate under a different name.

Remark 1.2. On a basis state, $P(\varphi)$ produces a global phase, which no measurement can see. On a superposition it produces a relative phase, which changes everything. We return to this difference in Section 2.2. It is the mechanism behind every algorithm in Lectures 7 to 9.

1.3 The Pauli Relations

The matrices I, X, Y, Z are the *Paulis*, and I is an honorary member. In Lecture 4, Block 3 we showed that they are Hermitian and unitary at the same time. Therefore they square to the identity, and their eigenvalues are ± 1. Learn their multiplication table by heart, so that you do not have to derive it each time.

The Pauli relations.

$$X^2 = Y^2 = Z^2 = I^2 = I, \qquad \sigma_j^\dagger = \sigma_j, \qquad \text{eigenvalues } \pm 1.$$

$$\textbf{Cyclic:} \quad XY = \mathrm{i}Z, \qquad YZ = \mathrm{i}X, \qquad ZX = \mathrm{i}Y.$$

$$\textbf{Anticyclic:} \quad YX = -\mathrm{i}Z, \qquad ZY = -\mathrm{i}X, \qquad XZ = -\mathrm{i}Y.$$

Distinct Paulis anticommute: $\sigma_j\sigma_k = -\sigma_k\sigma_j \quad (j \neq k,\ j,k \geq 1)$, and $XYZ = \mathrm{i}\,I$.

The importance of the sign

The products XZ and ZX differ by a sign. We have $ZX = +\mathrm{i}Y$ but $XZ = -\mathrm{i}Y$. Write the cyclic order $X \to Y \to Z \to X$ at the top of your page. A product that follows the arrow gets $+\mathrm{i}$. A product that goes against it gets $-\mathrm{i}$.
If you reverse this rule, the sign error spreads through every conjugation calculation in Block 2. Such errors stay invisible until an algorithm returns the wrong answer.

Remark 1.3 (the Pauli group)**.** A product of Paulis is a Pauli times a factor from $\{\pm 1, \pm\mathrm{i}\}$. So the set

$$\mathcal{P}_1 = \{\mathrm{i}^k\sigma : k \in \{0,1,2,3\},\ \sigma \in \{I,X,Y,Z\}\}$$

is a group of order 16. It is the *Pauli group* on one qubit. On n qubits, $\mathcal{P}_n$ contains the 4^n tensor strings $P_1 \otimes \cdots \otimes P_n$ with each $P_j \in \{I,X,Y,Z\}$, again times a phase in $\{\pm 1, \pm\mathrm{i}\}$.

Two Pauli strings either commute or anticommute. No other case occurs. Which one happens depends on the number of positions where the two strings hold *different* non-identity Paulis. An odd number gives anticommutation. This small fact drives all of Block 2.

1.4 Everything Is a Sum of Paulis

We can take a matrix apart in two ways, and we need both.

Taking a matrix apart by outer products

We can write any 2×2 matrix by picking out its entries one at a time:

$$M = \begin{bmatrix} a & b \\ c & d \end{bmatrix} = a\,|0\rangle\langle 0| + b\,|0\rangle\langle 1| + c\,|1\rangle\langle 0| + d\,|1\rangle\langle 1|\,. \tag{2}$$

This works because $|0\rangle\langle 0| = \left[\begin{smallmatrix} 1 & 0 \\ 0 & 0 \end{smallmatrix}\right]$ and $|0\rangle\langle 1| = \left[\begin{smallmatrix} 0 & 1 \\ 0 & 0 \end{smallmatrix}\right]$, and the same for the other two.

In Lecture 4 we found outer-product expansions by finding an eigenbasis. This expansion needs no properties of M at all. It also works for many qubits. For example, $|00\rangle\langle 11|$ is the 4×4 matrix with a single 1 in the appropriate corner.

Taking a matrix apart by Paulis

Each of those four outer products is itself a combination of Paulis:

$$|0\rangle\langle 0| = \frac{I+Z}{2}, \qquad |1\rangle\langle 1| = \frac{I-Z}{2}, \qquad |0\rangle\langle 1| = \frac{X+\mathrm{i}Y}{2}, \qquad |1\rangle\langle 0| = \frac{X-\mathrm{i}Y}{2}. \tag{3}$$

The first two follow at once. For the third, use $|1\rangle\langle 0| = X\,|0\rangle\langle 0|$, which holds because $X\,|0\rangle = |1\rangle$. Then

$$|1\rangle\langle 0| = X \cdot \frac{I+Z}{2} = \frac{X+XZ}{2} = \frac{X-\mathrm{i}Y}{2},$$

where the last step uses the anticyclic relation $XZ = -\mathrm{i}Y$. The fourth follows by taking the adjoint.

Remember the factors of half

Do not write $|0\rangle\langle 0| = I + Z$. That is wrong by a factor of two. The error is easy to miss, because the result still looks like a projector. Use the trace to test it. We have $\operatorname{tr}|0\rangle\langle 0| = 1$, but $\operatorname{tr}(I + Z) = 2$.

If we substitute (3) into (2), we get the statement that we want.

Proposition 1.4 (the Pauli basis)**.** *Every 2×2 matrix can be written in exactly one way as*

$$M = c_0 I + c_1 X + c_2 Y + c_3 Z, \qquad c_j = \tfrac{1}{2}\operatorname{tr}(\sigma_j M).$$

In detail, $c_0 = \frac{a+d}{2}$, $c_1 = \frac{b+c}{2}$, $c_2 = \frac{\mathrm{i}(b-c)}{2}$ and $c_3 = \frac{a-d}{2}$. The matrix M is Hermitian exactly when all four c_j are real.

Proof. The substitution above gives existence. For the coefficient formula, note that $\operatorname{tr}(\sigma_j\sigma_k) = 2\delta_{jk}$. The diagonal cases hold because $\sigma_j^2 = I$. The other cases hold because a product of two distinct Paulis is $\pm\mathrm{i}$ times the third, which has trace zero. Therefore $\operatorname{tr}(\sigma_k M) = \sum_j c_j \operatorname{tr}(\sigma_k\sigma_j) = 2c_k$.

Four matrices that are orthogonal in this sense are linearly independent. The space of 2×2 matrices has dimension four. So the four Paulis are a basis and the expansion is unique. If $M = M^\dagger$, take adjoints term by term and use $\sigma_j^\dagger = \sigma_j$. This gives $\sum_j c_j^* \sigma_j = \sum_j c_j \sigma_j$, and uniqueness forces $c_j^* = c_j$. □

Theorem 1.5 (the n-qubit Pauli basis)**.** *The 4^n Pauli strings $P = P_1 \otimes \cdots \otimes P_n$ with $P_j \in \{I, X, Y, Z\}$ form a basis for the $2^n \times 2^n$ matrices. Any such matrix is $M = \sum_P c_P P$ with $c_P = 2^{-n}\operatorname{tr}(PM)$. The matrix M is Hermitian exactly when every c_P is real.*

Proof. Take the tensor product of n copies of Proposition 1.4. Every matrix unit $|x\rangle\langle y|$, for bit strings x and y, factorises as $\bigotimes_l |x_l\rangle\langle y_l|$. Each factor expands in $\{I, X, Y, Z\}$ by (3). Multiply out and you get M as a combination of Pauli strings.

For the coefficients, we use $\operatorname{tr}(P \otimes Q) = \operatorname{tr} P \operatorname{tr} Q$ from Lecture 4, Block 4. Therefore $\operatorname{tr}(PP') = 2^n\delta_{PP'}$ for Pauli strings, and the argument of Proposition 1.4 runs unchanged. □

Example 1.6.

$$|00\rangle\langle 00| = |0\rangle\langle 0| \otimes |0\rangle\langle 0| = \frac{I+Z}{2} \otimes \frac{I+Z}{2} = \frac{1}{4}\big(I \otimes I + I \otimes Z + Z \otimes I + Z \otimes Z\big).$$

That is four Pauli strings, which the count $4^n = 4^2 = 16$ allows. The projector onto a single two-qubit basis state is a sum of four commuting Pauli strings, each with coefficient $\frac{1}{4}$. This is how we measure such projectors in practice. We measure $Z \otimes I$ and $I \otimes Z$, and then we combine the outcomes.

Theorem 1.5 and the spectral theorem together form the whole proof of universality in Block 3. Every unitary is $\mathrm{e}^{\mathrm{i}H}$ for a Hermitian H. Every Hermitian H is a real combination of Pauli strings. In Section 3.1 we then build a circuit for $\mathrm{e}^{\mathrm{i}\theta P}$ for any Pauli string P, from CNOT gates and one rotation. We now have the first ingredient.

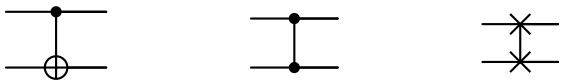


Figure 2: From left to right: CNOT with the control on top and the target below, then CZ, then SWAP. The CZ symbol is symmetric on purpose. See Remark 1.7.

1.5 Two-Qubit Gates

Three two-qubit gates do almost all the work.
In outer-product form, with the first qubit as control,

$$\text{CNOT} = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes X, \qquad \text{CZ} = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes Z. \tag{4}$$

As matrices in the basis $|00\rangle, |01\rangle, |10\rangle, |11\rangle$,

$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \qquad \text{CZ} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, \qquad \text{SWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

All three matrices are real and symmetric, so they are Hermitian. All three square to the identity, so they are unitary and are their own inverses. On basis states, CNOT is the map $|x\rangle|y\rangle \mapsto |x\rangle|y \oplus x\rangle$, which permutes the four basis vectors. Every permutation matrix has exactly one 1 in each row and in each column, and that is the shape you see above.

If instead the second qubit is the control, swap the roles of x and y:

$$\text{CNOT}_{2\to 1} = I \otimes |0\rangle\langle 0| + X \otimes |1\rangle\langle 1| = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

Remark 1.7 (CZ has no control)**.** The form $|0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes Z$ suggests that the first qubit controls the second. But the matrix $\text{diag}(1, 1, 1, -1)$ does not change when you exchange the two qubits. The gate CZ applies a -1 exactly when *both* qubits are $|1\rangle$, and that condition treats them alike. This is why the symbol in Figure 2 has two dots. In Lecture 7 we use the same observation for the controlled phase gates of the Fourier transform.

Proposition 1.8 (three identities)**.**

$$\textit{(i)} \quad \text{CNOT}_{1\to 2} = (I \otimes H)\,\text{CZ}\,(I \otimes H), \qquad \textit{(ii)} \quad (H \otimes H)\,\text{CNOT}_{1\to 2}\,(H \otimes H) = \text{CNOT}_{2\to 1},$$

$$\textit{(iii)} \quad \text{SWAP} = \text{CNOT}_{1\to 2}\,\text{CNOT}_{2\to 1}\,\text{CNOT}_{1\to 2}.$$

Proof. For (i), conjugate the second tensor factor of (4) by H. This leaves I alone and sends Z to $HZH = X$, so CZ becomes CNOT.

For (ii), use (i) to get $(H \otimes H)\text{CNOT}(H \otimes H) = (H \otimes I)\text{CZ}(H \otimes I)$. Conjugating the *first* factor turns CZ into a CNOT with the roles of the two qubits exchanged. The symmetry of CZ in Remark 1.7 is what makes this work.

For (iii), track a basis state: $|x, y\rangle \mapsto |x, x \oplus y\rangle \mapsto |x \oplus (x \oplus y), x \oplus y\rangle = |y, x \oplus y\rangle \mapsto |y, x \oplus y \oplus y\rangle = |y, x\rangle$. □

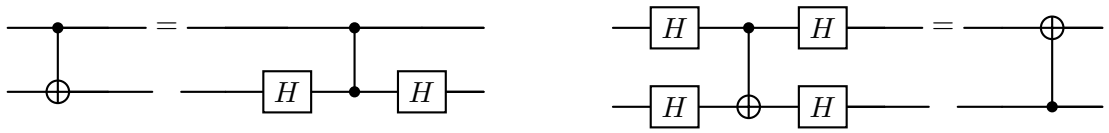


Figure 3: Identities (i) and (ii). On the right, Hadamards on both wires reverse which qubit is the control. This is the simplest case of a theme that dominates Block 2. Conjugating a gate changes what it does to the Paulis, and so it changes the gate.

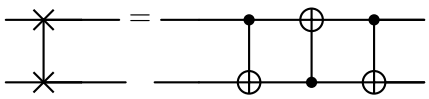


Figure 4: Identity (iii), the three-CNOT swap. Remember it as a cost and not only as an identity. On hardware where only neighbouring qubits interact, moving a qubit one step across the chip costs three CNOT gates. In Lecture 7, Section 3.8 we charge the quantum Fourier transform for exactly this cost.

1.6 Bell States and Correlations

Two gates are enough to produce entanglement. Apply H to the first qubit of $|00\rangle$ and then apply CNOT:

$$|00\rangle \xrightarrow{H\otimes I} \frac{|00\rangle + |10\rangle}{\sqrt{2}} \xrightarrow{\text{CNOT}} \frac{|00\rangle + |11\rangle}{\sqrt{2}} = \left|\Phi^+\right\rangle .$$

In Lecture 4 we showed that this is not a product state.

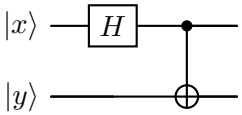


Figure 5: The Bell circuit. On the four inputs $|x, y\rangle$ it produces the four Bell states $|\Phi^{\pm}\rangle = (|00\rangle \pm |11\rangle)/\sqrt{2}$ and $|\Psi^{\pm}\rangle = (|01\rangle \pm |10\rangle)/\sqrt{2}$. These are an orthonormal basis of $\mathbb{C}^4$. The circuit is unitary, so we can run it backwards to measure in that basis.

The Bell states and the computational basis states are both orthonormal bases. The difference between them is how the two qubits are correlated, and that difference shows up only when you compare two measurement bases.

Proposition 1.9 (the Bell state agrees in two bases)**.**

$$\left|\Phi^+\right\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}} = \frac{|{+}{+}\rangle + |{-}{-}\rangle}{\sqrt{2}}.$$

The two outcomes always agree when both qubits are measured in the Z basis. They always agree when both are measured in the X basis.

Proof. Expand the right-hand side. With $|{+}{+}\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$ and $|{-}{-}\rangle = \frac{1}{2}(|00\rangle - |01\rangle - |10\rangle + |11\rangle)$, the two middle terms cancel and the sum is $|00\rangle + |11\rangle$. Divide by $\sqrt{2}$.

For the measurements, read each form in turn. The Z form is supported on $|00\rangle$ and $|11\rangle$, so the two outcomes are equal. The X form is supported on $|{+}{+}\rangle$ and $|{-}{-}\rangle$, so the two outcomes are equal again. □

Proposition 1.10 (no product state does this)**.** *No product state $|\alpha\rangle \otimes |\beta\rangle$ gives outcomes that agree with certainty in the Z basis and also with certainty in the X basis.*

Proof. Suppose the Z outcomes always agree. Each qubit is measured on its own, so each factor must give a definite Z outcome. That forces $|\alpha\rangle$ and $|\beta\rangle$ to be $|0\rangle$ or $|1\rangle$, and it forces them to be the same one. Take $|\alpha\rangle = |\beta\rangle = |0\rangle$.

Now measure both in the X basis. Since $|0\rangle = (|+\rangle + |-\rangle)/\sqrt{2}$, each qubit returns $+$ or $-$ with probability $\frac{1}{2}$, and the two qubits are independent. The outcomes therefore agree only half the time. The case $|\alpha\rangle = |\beta\rangle = |1\rangle$ is the same. □

Remark 1.11. This is the smallest statement of a fact that we use again and again in the rest of the subject. Agreement in one basis is cheap, and two classical coins can do it. Agreement in two different bases at once needs entanglement. Push the same argument further, with three bases instead of two, and you get the Bell inequalities.

1.7 More Than Two Qubits: GHZ States and Partial Measurement

The Bell circuit extends to n qubits in more than one way. Here are two circuits on four qubits. The first spreads the control out to every other wire. The second passes it along a chain.

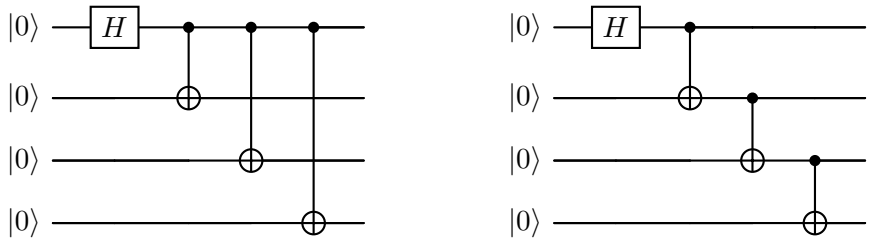


Figure 6: Two circuits for the four-qubit GHZ state. On the left the first qubit controls all three others, which is called fan-out. On the right the control is passed along a chain. On the input $|0000\rangle$ they give the same output.

Proposition 1.12 (the GHZ state, two ways)**.** *Both circuits of Figure 6 map $|0000\rangle$ to*

$$|\mathrm{GHZ}\rangle = \frac{|0000\rangle + |1111\rangle}{\sqrt{2}}.$$

They are nevertheless different unitaries.

Proof. The Hadamard gives $(|0\rangle + |1\rangle)/\sqrt{2}$ on the first wire, so the state is $(|0000\rangle + |1000\rangle)/\sqrt{2}$. In the fan-out circuit each CNOT copies the first bit onto one further wire, and after three of them the second term is $|1111\rangle$. Now take the chain circuit. The first CNOT makes the second bit equal to the first. The second passes that value to the third wire, and the third passes it to the fourth. The second term is again $|1111\rangle$. The first term is $|0000\rangle$ in both cases, because every control is 0.

To see that the two circuits differ, run both on $|0100\rangle$:

$$\text{fan-out:}\quad \frac{|0100\rangle + |1011\rangle}{\sqrt{2}}, \qquad \text{chain:}\quad \frac{|0111\rangle + |1000\rangle}{\sqrt{2}}.$$

These states are orthogonal, so the two unitaries are not equal. □

One input is not enough

Two circuits that agree on one input can still be different operators. A circuit on n qubits is fixed by its action on all 2^n computational basis states. So you must check all of them, or find one where the two disagree. The Fact of Lecture 4, Block 2 is the positive half of this statement. The proposition above is the warning that goes with it.

Measuring one qubit out of many

In Lecture 4, Block 3 we described a measurement as a set of projectors $\{P_k\}$ with $\sum_k P_k = I$. The outcome k has probability $p(k) = \|P_k|\psi\rangle\|^2$, and the state afterwards is $P_k|\psi\rangle/\sqrt{p(k)}$. Nothing in that description requires the projectors to act on every qubit. To measure the first qubit alone in the Z basis, take

$$P_0 = |0\rangle\langle 0| \otimes I \otimes I \otimes I, \qquad P_1 = |1\rangle\langle 1| \otimes I \otimes I \otimes I.$$

The other three qubits are untouched, and the formula tells you what is left of them.

Example 1.13 (the Z measurement destroys the entanglement)**.** Apply P_0 to $|\mathrm{GHZ}\rangle$. Only the $|0000\rangle$ term survives, so $P_0|\mathrm{GHZ}\rangle = |0000\rangle/\sqrt{2}$ and $p(0) = \frac{1}{2}$. After normalisation the remaining three qubits are in $|000\rangle$. The outcome 1 has probability $\frac{1}{2}$ and leaves $|111\rangle$. Both are product states, so one measurement removes all the entanglement from the other three qubits.

Example 1.14 (the X measurement preserves it)**.** Now measure the first qubit in the X basis instead, which means taking $P_\pm = |\pm\rangle\langle\pm| \otimes I \otimes I \otimes I$. Substitute $|0\rangle = (|+\rangle + |-\rangle)/\sqrt{2}$ and $|1\rangle = (|+\rangle - |-\rangle)/\sqrt{2}$ into the first factor:

$$|\mathrm{GHZ}\rangle = \frac{1}{2}\Big[|+\rangle\big(|000\rangle + |111\rangle\big) + |-\rangle\big(|000\rangle - |111\rangle\big)\Big].$$

Each outcome has probability $\frac{1}{2}$. The three remaining qubits are left in $(|000\rangle \pm |111\rangle)/\sqrt{2}$, which is a three-qubit GHZ state and is still entangled.

The two examples use the same state and the same single qubit. The basis you measure in decides what is left behind. In the Z basis the rest of the register collapses to a product state. In the X basis it keeps its entanglement and only shrinks by one qubit.
Remember this effect for two reasons. First, it is the first case where a measurement on part of a register does something interesting to the rest. Second, in Lecture 6 we meet the same effect from the other side. There, the garbage register acts as an accidental measurement, and it destroys the interference that the algorithm needs.

Complicating matters of entanglement

It is tempting to conclude that entanglement is the resource behind the speed of quantum computers. If that is true, then two gates give us the whole subject. In Block 3 we show that this conclusion is false.
The gates H and CNOT belong to a class called the Cliffords. Take any circuit built from Clifford gates, run on computational basis states and measured in the computational basis. The *Gottesman–Knill theorem* says that a laptop can simulate it in polynomial time. The amount of entanglement does not change this.
Entanglement is necessary for a speed-up, but it is far from sufficient. In Block 2 we add what is missing from $\{H, \mathrm{CNOT}\}$. In Block 3 we explain why its absence is fatal.

2 Block 2: Rotations, Cliffords, and Controlled Gates

By the end of this block you can...

- Explain why every unitary is e^{iH} for some Hermitian H, and compute $e^{i\theta P}$ for a Pauli P in closed form.
- Define R_x, R_y, R_z, state the convention for the sign of the angle, and relate them to X, S, T and H.
- Tell global phase from relative phase, and say when a global phase stops being global.
- State the defining property of the Clifford group and use the conjugation tables for H, S and CNOT.
- Build a controlled phase gate from two CNOT gates and three phase gates, and a general controlled-U from two CNOT gates and three single-qubit gates.
- Construct the Toffoli gate and gates with several controls, and state their cost.
- Derive phase kickback for CNOT acting on $|+\rangle\,|-\rangle$.

2.1 Every Unitary Is an Exponential

In Lecture 4 we proved two facts, and now we combine them. A unitary U is normal, so it has an orthonormal eigenbasis $\{|h_j\rangle\}$. Its eigenvalues have modulus one, so you can write them as e^{ih_j} with h_j real:

$$U = \sum_j e^{ih_j}\,|h_j\rangle\langle h_j|\,.$$

A Hermitian operator is also normal, and its eigenvalues are real:

$$H = \sum_j h_j\,|h_j\rangle\langle h_j|\,.$$

The same list of real numbers h_j appears in both. In one case they are the eigenvalues. In the other they are the phases of the eigenvalues. This is a correspondence and not a coincidence.

Theorem 2.1 (unitary and Hermitian)**.** *For every Hermitian H, the operator e^{iH} is unitary. Every unitary U can be written as e^{iH} for some Hermitian H. Here e^{iH} means the operator function of Lecture 4, Block 4. If $H = \sum_j h_j\,|h_j\rangle\langle h_j|$ then $e^{iH} = \sum_j e^{ih_j}\,|h_j\rangle\langle h_j|$.*

Proof. Take a Hermitian H with real eigenvalues h_j. The operator $e^{iH} = \sum_j e^{ih_j}\,|h_j\rangle\langle h_j|$ has eigenvalues of modulus one in an orthonormal basis. Therefore $(e^{iH})^\dagger e^{iH} = \sum_j |e^{ih_j}|^2\,|h_j\rangle\langle h_j| = I$.

Now take a unitary $U = \sum_j e^{ih_j}\,|h_j\rangle\langle h_j|$. Each h_j can be chosen real because $|e^{ih_j}| = 1$. Set $H = \sum_j h_j\,|h_j\rangle\langle h_j|$. This H is Hermitian because the h_j are real, and $e^{iH} = U$. □

Remark 2.2. The exponential of a matrix is defined by its Taylor series, $e^A = \sum_{k\geq 0} A^k/k!$. For normal A the two definitions agree. Powers of $\sum_j a_j\,|v_j\rangle\langle v_j|$ are $\sum_j a_j^k\,|v_j\rangle\langle v_j|$, because $|v_i\rangle\langle v_i|\,|v_j\rangle\langle v_j| = \delta_{ij}\,|v_i\rangle\langle v_i|$ removes all the cross terms. Sum the series term by term and you get $\sum_j e^{a_j}\,|v_j\rangle\langle v_j|$. Note that H is not unique. Adding 2π to any h_j leaves U unchanged.

Proposition 2.3 (Pauli exponentials in closed form)**.** *Let P be any Pauli string, so that $P^2 = I$. Then*

$$e^{-i\theta P/2} = \cos\tfrac{\theta}{2}\,I - i\sin\tfrac{\theta}{2}\,P.$$

Proof. Split the Taylor series into even and odd powers and use $P^{2k} = I$ and $P^{2k+1} = P$:

$$e^{-i\theta P/2} = \sum_k \frac{(-i\theta/2)^{2k}}{(2k)!} I + \sum_k \frac{(-i\theta/2)^{2k+1}}{(2k+1)!} P = \cos\tfrac{\theta}{2}\, I - i \sin\tfrac{\theta}{2}\, P.$$

Here is a second route. The operator P has eigenvalues ± 1, so $e^{-i\theta P/2}$ has eigenvalues $e^{\mp i\theta/2}$ on the same eigenspaces. The projectors $P_\pm = (I \pm P)/2$ then reassemble the formula. □

The single-qubit case defines the *rotation gates*:

$$R_x(\theta) = e^{-i\theta X/2}, \qquad R_y(\theta) = e^{-i\theta Y/2}, \qquad R_z(\theta) = e^{-i\theta Z/2} = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}. \tag{5}$$

Sign convention. We use the minus sign in the exponent. Then $R_{\hat{n}}(\theta)$ rotates the Bloch vector by $+\theta$ about the axis $\hat{n}$, in the right-handed sense. Some sources write $e^{+i\theta X/2}$ instead, which is our $R_x(-\theta)$. Pick one convention and never mix them.
Angle halving. The factor $\theta/2$ matters. A rotation by $\theta = 2\pi$ gives $R_x(2\pi) = -I$ and not I. A full turn of the Bloch sphere returns the state to itself with a minus sign. That sign is global and invisible until we control the gate (see Section 2.2).

Example 2.4 (the familiar gates as rotations)**.** We have $R_z(\pi) = \mathrm{diag}(e^{-i\pi/2}, e^{i\pi/2}) = -iZ$, so Z is a half-turn about $\hat{z}$ up to a phase. In the same way $X = i\,R_x(\pi)$, $S = e^{i\pi/4} R_z(\pi/2)$ and $T = e^{i\pi/8} R_z(\pi/4)$. The Hadamard is a half-turn about the diagonal axis $(\hat{x} + \hat{z})/\sqrt{2}$, because $H = (X + Z)/\sqrt{2}$ and $H^2 = I$.

The phase gate of (1) is $P(\varphi) = e^{i\varphi/2} R_z(\varphi)$. It differs from R_z only by a global phase. The two are interchangeable, except inside a controlled gate.

2.2 Global and Relative Phase

A *global phase* is a factor $e^{i\gamma}$ that multiplies a whole state. In Lecture 4 we showed that no measurement in any basis can detect it, because $|\langle v|e^{i\gamma}\psi\rangle|^2 = |\langle v|\psi\rangle|^2$. A *relative* phase sits between the terms of a superposition and is easy to detect. The states $|+\rangle$ and $|-\rangle$ differ only by a relative phase, and they are orthogonal.

A control turns a global phase into a relative one

This is the most common source of sign errors in circuit work. Let $V = e^{i\gamma} U$. Then V and U are the same gate for every practical purpose. But $\Lambda(V)$ and $\Lambda(U)$ are different gates:

$$\Lambda(V) = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes e^{i\gamma} U \neq |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes U = \Lambda(U).$$

The phase lands only on the branch where the control is $|1\rangle$. It is therefore relative between the two branches.
Here is a concrete case. The gates Z and $R_z(\pi) = -iZ$ are interchangeable. But $\Lambda(Z) = \mathrm{CZ}$ and $\Lambda(R_z(\pi))$ differ by a phase gate on the control. Before you drop a phase, check that nobody will control the gate.

Remark 2.5. This is not only a hazard. Read the other way round, it is the central mechanism of the subject. In Lecture 8, phase estimation takes a phase that is invisible on an eigenstate and makes it visible. It does this by controlling the gate on an ancilla in superposition. In Section 2.5 we look at the two-qubit case.

2.3 The Clifford Group

Paulis are cheap and, on their own, dull. The interesting structure is the set of gates that *rearrange* them.

Definition 2.6 (Clifford group)**.** A unitary U on n qubits is a *Clifford* when conjugation by U maps Pauli strings to Pauli strings:

$$U\mathcal{P}_nU^\dagger = \mathcal{P}_n, \qquad \text{that is,} \qquad UPU^\dagger \in \mathcal{P}_n \ \text{ for every } P \in \mathcal{P}_n.$$

The Cliffords form a group $\mathcal{C}_n$, which contains the Paulis themselves.

Conjugation, which means the map $P \mapsto UPU^\dagger$, is the natural operation here. The reason is a one-line calculation that we use again and again. If P has eigenvalue λ on $|v\rangle$, then $UPU^\dagger$ has eigenvalue λ on $U|v\rangle$. So when we conjugate an operator, we ask what it looks like in the rotated basis.

The conjugation tables. The gates $\{H, S, \text{CNOT}\}$ generate the Clifford group. These are their actions.

	$X \mapsto$	$Y \mapsto$	$Z \mapsto$
H	Z	$-Y$	X
S	Y	$-X$	Z
X	X	$-Y$	$-Z$
Z	$-X$	$-Y$	Z

CNOT, control c, target t	
$X_c \mapsto X_cX_t$	$X_t \mapsto X_t$
$Z_c \mapsto Z_c$	$Z_t \mapsto Z_cZ_t$
$Y_c \mapsto Y_cX_t$	$Y_t \mapsto Z_cY_t$

Mnemonic: *CNOT copies X forwards and Z backwards.*

Spot-checks. Direct multiplication gives $HXH = Z$ and $HZH = X$. Then $HYH = H(\mathrm{i}XZ)H = \mathrm{i}(HXH)(HZH) = \mathrm{i}ZX = \mathrm{i}(\mathrm{i}Y) = -Y$.

For S, compute $SXS^\dagger = \left[\begin{smallmatrix}1&0\\0&\mathrm{i}\end{smallmatrix}\right]\left[\begin{smallmatrix}0&1\\1&0\end{smallmatrix}\right]\left[\begin{smallmatrix}1&0\\0&-\mathrm{i}\end{smallmatrix}\right] = \left[\begin{smallmatrix}0&-\mathrm{i}\\\mathrm{i}&0\end{smallmatrix}\right] = Y$, and note that S commutes with Z. For the Paulis themselves, anticommutation gives $XZX^\dagger = XZX = -ZX^2 = -Z$.

For CNOT, use $\text{CNOT}\,|x, y\rangle = |x, x \oplus y\rangle$. Flipping the control before the gate has the same effect as flipping both wires after it, which gives $X_c \mapsto X_cX_t$. A phase $(-1)^y$ after the gate equals the phase $(-1)^{x\oplus y}$ before it, which gives $Z_t \mapsto Z_cZ_t$. □

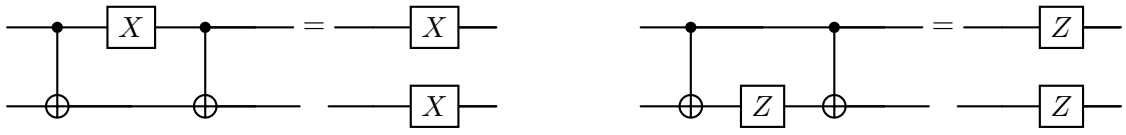


Figure 7: The mnemonic as circuits. On the left, an X on the control, conjugated by CNOT, becomes X on *both* wires. On the right, a Z on the target becomes Z on both. Bit-flip errors therefore spread from control to target, and phase-flip errors spread the other way. We use this fact again in the lectures on error correction.

Example 2.7 (trading one rotation for another)**.** Suppose your hardware offers $R_x(\theta)$ for any θ, but you need $R_z(\theta)$. Conjugation solves the problem. The unitary and its Hermitian generator share an eigenbasis, so conjugating the unitary is the same as conjugating the generator:

$$H\,R_x(\theta)\,H^\dagger = H\,\mathrm{e}^{-\mathrm{i}\theta X/2}\,H^\dagger = \mathrm{e}^{-\mathrm{i}\theta(HXH^\dagger)/2} = \mathrm{e}^{-\mathrm{i}\theta Z/2} = R_z(\theta).$$

The middle step needs a closer look. The rule $He^{A}H^{\dagger} = e^{HAH^{\dagger}}$ holds because conjugation distributes over every term of the Taylor series, and the intermediate $H^{\dagger}H$ pairs cancel. We use this move again and again. It is the engine of Section 3.1.

2.4 Controlled Gates and How to Build Them

For any single-qubit unitary U, the controlled gate is

$$\Lambda(U) = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes U.$$

No hardware provides this directly for a general U. We must compile it from smaller gates. This compilation is one of the most useful constructions in the subject.

The easy case: a controlled phase

Let $U = P(\theta) = \mathrm{diag}(1, e^{i\theta})$. Then we can build $\Lambda(P(\theta))$ from two CNOT gates and three copies of $P(\theta/2)$.

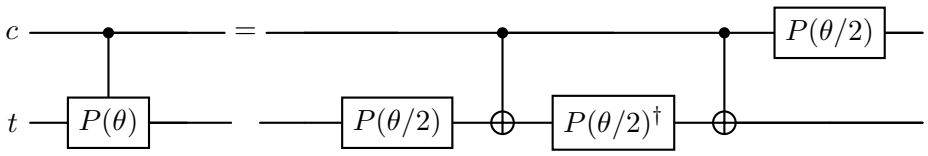


Figure 8: The controlled phase gate. In Lecture 7 we build the quantum Fourier transform entirely from controlled $R_k = P(2\pi/2^k)$ gates. We assemble each one this way, from two CNOT gates and three copies of $R_{k+1} = P(\pi/2^k)$, one of them inverted.

Verification. Track the phase collected on each basis state $|c, t\rangle$, and write $\phi = \theta/2$. The target collects $e^{i\phi}$ when $t = 1$, then $e^{-i\phi}$ when $t \oplus c = 1$. The control collects $e^{i\phi}$ when $c = 1$:

$$\begin{aligned} |00\rangle &: \ 1 \cdot 1 \cdot 1 = 1, & |01\rangle &: \ e^{i\phi} \cdot e^{-i\phi} \cdot 1 = 1, \\ |10\rangle &: \ 1 \cdot e^{-i\phi} \cdot e^{i\phi} = 1, & |11\rangle &: \ e^{i\phi} \cdot 1 \cdot e^{i\phi} = e^{i\theta}. \end{aligned}$$

So the circuit is $\mathrm{diag}(1, 1, 1, e^{i\theta}) = \Lambda(P(\theta))$. Note that this matrix is symmetric in c and t. Like CZ, a controlled phase has no distinguished control. □

The general case

Every single-qubit unitary can be written as $U = e^{i\alpha} R_z(\beta) R_y(\gamma) R_z(\delta)$, which is three rotations and a phase. This is the statement that the rotation group of the Bloch sphere has three dimensions. From that decomposition we can always extract three gates whose product is the identity, but whose X-interleaved product is U.

Theorem 2.8 (controlled-U from two CNOT gates)**.** *For any single-qubit unitary U there are single-qubit unitaries A, B, C and a real number α with*

$$ABC = I \qquad \text{and} \qquad U = e^{i\alpha} AXBXC.$$

The circuit of Figure 9 then realises $\Lambda(U)$.

Proof. Write $U = e^{i\alpha} R_z(\beta) R_y(\gamma) R_z(\delta)$ and set

$$A = R_z(\beta) R_y(\tfrac{\gamma}{2}), \qquad B = R_y(-\tfrac{\gamma}{2}) R_z(-\tfrac{\delta+\beta}{2}), \qquad C = R_z(\tfrac{\delta-\beta}{2}).$$

Then $ABC = R_z(\beta)R_z\big(-\frac{\delta+\beta}{2} + \frac{\delta-\beta}{2}\big) = R_z(\beta)R_z(-\beta) = I$.

For the other product, use $XR_y(\vartheta)X = R_y(-\vartheta)$ and $XR_z(\vartheta)X = R_z(-\vartheta)$. Both come from the conjugation tables, because X anticommutes with Y and with Z. They give $XBX = R_y(\frac{\gamma}{2})R_z(\frac{\delta+\beta}{2})$, and therefore

$$AXBXC = R_z(\beta)R_y(\tfrac{\gamma}{2})R_y(\tfrac{\gamma}{2})R_z(\tfrac{\delta+\beta}{2})R_z(\tfrac{\delta-\beta}{2}) = R_z(\beta)R_y(\gamma)R_z(\delta) = \mathrm{e}^{-\mathrm{i}\alpha}U.$$

In the circuit, a control of $|0\rangle$ leaves the two CNOT gates idle, and the target sees $ABC = I$. A control of $|1\rangle$ makes the target see $AXBXC$, and the phase gate on the control supplies $\mathrm{e}^{\mathrm{i}\alpha}$. □

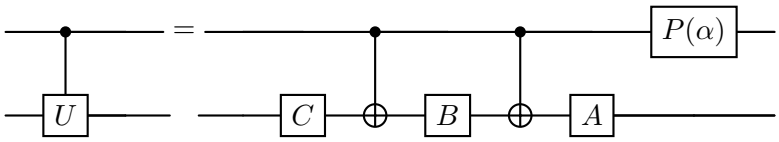


Figure 9: A general controlled-U. It uses two CNOT gates, three single-qubit gates on the target, and one phase gate on the control. Read it from right to left as a matrix product and from left to right as a circuit. The gate applied first is C.

Three controls and more

The *Toffoli* gate $\Lambda^2(X)$ flips its target when both controls are $|1\rangle$. It is the workhorse of Block 4, because a preset target makes it compute classical logic:

$$\text{TOFFOLI}: \ |a,b,c\rangle \mapsto |a,b,\, c \oplus (a \wedge b)\rangle\,, \qquad \text{so} \qquad |a,b,0\rangle \mapsto |a,b,a\wedge b\rangle\,, \quad |a,b,1\rangle \mapsto \big|a,b,\overline{a\wedge b}\big\rangle\,.$$

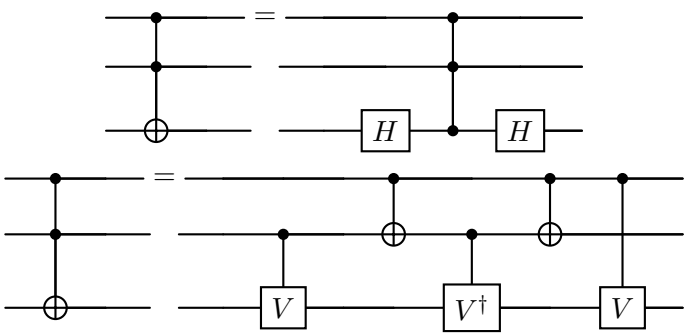


Figure 10: On the left, Toffoli and CCZ differ by Hadamards on the target, in the same way that CNOT and CZ do. On the right is the standard way to split a doubly controlled gate into singly controlled ones, where $V^2 = U$. For $U = X$ you take $V = \sqrt{X}$. Expand each controlled-V by Theorem 2.8 and you get the familiar six-CNOT Toffoli.

Remark 2.9 (costs worth remembering)**.** A Toffoli costs 6 CNOT gates and a few single-qubit gates. A gate with $n-1$ controls, written $\Lambda^{n-1}(X)$, costs $\mathcal{O}(n)$ Toffoli gates when you can borrow $n-2$ ancilla qubits in state $|0\rangle$. The pattern computes and then uncomputes, and we meet it again in Block 4. Without ancillas the cost is $\mathcal{O}(n^2)$. The general lesson is that extra qubits buy a lower gate count. This trade-off appears everywhere.

2.5 Phase Kickback, the Two-Qubit Case

Here is the smallest case of an effect that we use at the start of Lecture 7 and throughout Lecture 8. We prepare the target of a CNOT in $|-\rangle$ rather than $|0\rangle$. Recall that $X|-\rangle = -|-\rangle$.

Proposition 2.10 (kickback)**.** *For $x \in \{0,1\}$ we have* CNOT $|x\rangle|-\rangle = (-1)^x |x\rangle|-\rangle$*, and therefore*

$$\text{CNOT}\,|+\rangle|-\rangle = |-\rangle|-\rangle\,.$$

Proof. If $x = 0$, nothing happens. If $x = 1$, the target receives X, and $X|-\rangle = -|-\rangle$. So the state collects a factor -1 and the target returns unchanged. By linearity, apply this to $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$ and you get $(|0\rangle - |1\rangle)/\sqrt{2} \otimes |-\rangle = |-\rangle|-\rangle$. □

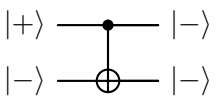


Figure 11: Phase kickback. The drawing shows the top wire controlling the bottom one, and the matrix says the same. Yet the state that changes is the control. The -1 that the target was going to collect moves onto the control instead. There it is a relative phase.

The general statement, which we need in Block 4 and in Lecture 8. If $U|u\rangle = \mathrm{e}^{\mathrm{i}\varphi}|u\rangle$, then

$$\Lambda(U)\;|c\rangle|u\rangle = \mathrm{e}^{\mathrm{i}c\varphi}|c\rangle|u\rangle\,, \qquad c \in \{0,1\}.$$

A controlled gate acting on an *eigenstate* of the target does nothing to the target. It writes the phase of the eigenvalue onto the control. Proposition 2.10 is the case $U = X$, $|u\rangle = |-\rangle$ and $\varphi = \pi$.

This is the exact sense in which the pitfall of Section 2.2 is also the main tool of the subject. The phase was global on $|u\rangle$ and invisible. When we control the gate, the phase becomes relative, and so we can read it.

Part II

Lecture 6: Universality and Computation

In Lecture 5 we assembled a toolbox of gates, identities, and the pictures that represent them. Two questions remain before we can start on the algorithms.

The first question is whether the toolbox is enough. A quantum computer must be able to apply an arbitrary unitary, and we gave it only $R_x(\theta)$, H, S and CNOT to do so. In Block 3 we show that these gates suffice for every unitary, to any accuracy. The proof uses only the spectral theorem, the Pauli basis and one circuit.

The second question is where the oracles come from. In Lectures 7, 8 and 9 we always start by assuming that we have a unitary U_f that computes a classical function. Here we learn how to build it. On the way we meet three facts that every quantum programmer must know. You cannot copy an unknown state. You cannot throw away a scratch register. You can work around the second fact if you run the computation backwards.

3 Block 3: Universality

By the end of this block you can...

- Draw the circuit for $e^{-i\theta P/2}$ for any Pauli string P, and count its gates.
- Explain why $e^A e^B \neq e^{A+B}$ for A and B that do not commute, and state the Trotter–Suzuki remedy with its error scaling.
- Give the four-step proof that $\{R_x(\theta), H, S, \text{CNOT}\}$ is universal.
- Name three other universal gate sets and say what Solovay–Kitaev adds.
- State the counting argument that shows that almost every unitary needs exponentially many gates, and explain why this does not contradict universality.
- State the Gottesman–Knill theorem and draw the right conclusion about entanglement.

A short reason for this block

In this block we answer a question that we left open in Lecture 4. We said that gates are unitary. We then gave you five or six specific unitaries. So far, nothing rules out the possibility that the machine reaches only a thin, useless corner of the unitary group.
In fact, the machine reaches all of it. The argument is short, and it uses only material that you already have. The spectral theorem says that every unitary is e^{iH}. The Pauli basis says that every H is a real combination of Pauli strings. One circuit realises a single Pauli exponential. Trotter–Suzuki joins the pieces. That makes four steps, and we already saw three of them.
The honest caveat comes at the end of the block, and it matters more than the theorem. Universality says that every unitary can be reached. It does not say that reaching it is cheap. For almost all unitaries it is not.

3.1 A Pauli Exponential Is a Circuit

Everything depends on the following construction. We want a circuit for $e^{-i\theta P/2}$, where $P = P_1 \otimes \cdots \otimes P_n$ is a Pauli string. We start with the case where every P_j is Z or I.

Proposition 3.1 (the CNOT ladder)**.** *The circuit of Figure 12 implements* $e^{-i\theta\, Z\otimes Z\otimes Z\otimes Z/2}$*. It uses* $2(n-1)$ *CNOT gates and one* $R_z(\theta)$*.*

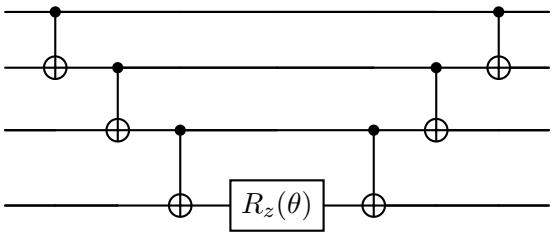


Figure 12: The CNOT ladder for $e^{-i\theta Z^{\otimes 4}/2}$. The descending CNOT gates collect the parity of the four bits onto the bottom wire. The rotation then applies a phase that depends on that parity. The ascending CNOT gates put everything back.

Proof. Take a computational basis state $|x_1x_2x_3x_4\rangle$. The first CNOT replaces x_2 by $x_1 \oplus x_2$. The second replaces x_3 by $x_1 \oplus x_2 \oplus x_3$. After the full descending ladder the bottom wire holds the parity $p = x_1 \oplus x_2 \oplus x_3 \oplus x_4$, and the other wires are unchanged.

Now $R_z(\theta) = \mathrm{diag}(e^{-i\theta/2}, e^{i\theta/2})$ and $(-1)^p$ is $+1$ for $p = 0$ and -1 for $p = 1$, so

$$R_z(\theta)\,|p\rangle = e^{-i\theta(-1)^p/2}\,|p\rangle\,.$$

The ascending ladder inverts the descending one, so it restores the original bit string. The net effect is multiplication by $e^{-i\theta(-1)^p/2}$.

Finally, $(-1)^p$ is the eigenvalue of $Z \otimes Z \otimes Z \otimes Z$ on $|x_1x_2x_3x_4\rangle$. So the circuit multiplies each basis state by the eigenvalue of $e^{-i\theta Z^{\otimes 4}/2}$, which is what that operator does. □

To get an arbitrary Pauli string, we conjugate the circuit. In Example 2.7 we saw that $Ve^{A}V^\dagger = e^{VAV^\dagger}$. So we take V to be a layer of single-qubit gates with $V(Z \otimes \cdots \otimes Z)V^\dagger = P$. If we apply V around the ladder, we get $e^{-i\theta P/2}$. The conjugation table supplies the gates that we need:

$$H\,Z\,H^\dagger = X, \qquad (SH)\,Z\,(SH)^\dagger = S\,X\,S^\dagger = Y, \qquad I\,Z\,I = Z,$$

and a wire with $P_j = I$ is left out of the ladder.

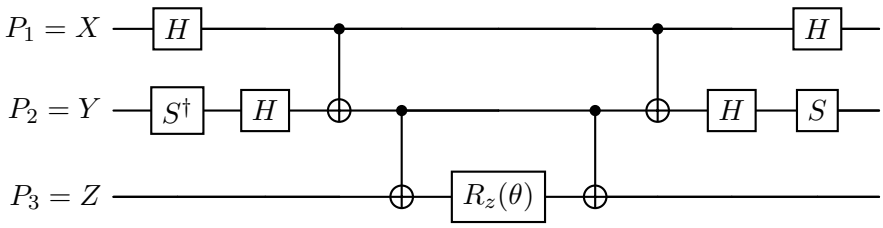


Figure 13: The circuit for $e^{-i\theta\,(X\otimes Y\otimes Z)/2}$. The outer layers rotate the Pauli on each wire into Z. The ladder and the rotation then do the work, and the layers are undone. A Pauli string of weight w costs $2(w-1)$ CNOT gates, one rotation, and at most four single-qubit gates per wire.

The point. For every Pauli string P and every angle θ, the unitary $e^{-i\theta P/2}$ has an *exact* circuit over $\{R_z(\theta), H, S, \mathrm{CNOT}\}$. Its size is linear in the number of qubits that P touches. Since $R_z(\theta) = H\,R_x(\theta)\,H$, the gate set $\{R_x(\theta), H, S, \mathrm{CNOT}\}$ works equally well.
This one family of gates is very expressive. It is also exactly what a physicist means by a

term in a Hamiltonian. We explain this coincidence in Remark 3.9.

3.2 Why You Cannot Just Multiply Exponentials

A general Hermitian H is a *sum* of Pauli strings. We want $\mathrm{e}^{\mathrm{i}H}$ to be the product of the matching circuits. For numbers, this is automatic. For matrices, it is false.

Proposition 3.2 (the obstruction is the commutator)**.** *For matrices A and B,*

$$\mathrm{e}^{A}\mathrm{e}^{B} = \mathrm{e}^{A+B} + \tfrac{1}{2}[A,B] + (\textit{terms of third order}).$$

So $\mathrm{e}^{A}\mathrm{e}^{B} = \mathrm{e}^{A+B}$ whenever $AB = BA$, and in general not otherwise.

Proof. Expand both sides to second order:

$$\mathrm{e}^{A}\mathrm{e}^{B} = \left(I + A + \tfrac{A^2}{2} + \cdots\right)\left(I + B + \tfrac{B^2}{2} + \cdots\right) = I + A + B + \tfrac{A^2}{2} + AB + \tfrac{B^2}{2} + \cdots,$$

$$\mathrm{e}^{A+B} = I + (A+B) + \tfrac{(A+B)^2}{2} + \cdots = I + A + B + \tfrac{A^2+AB+BA+B^2}{2} + \cdots.$$

Subtract. The second-order terms differ by $AB - \frac{1}{2}(AB+BA) = \frac{1}{2}[A,B]$. □

Paulis do not commute, which brings about more issues than you may thing

The matrices X and Y anticommute, so $[X,Y] = 2XY = 2\mathrm{i}Z \neq 0$ and $\mathrm{e}^{\mathrm{i}X}\mathrm{e}^{\mathrm{i}Y} \neq \mathrm{e}^{\mathrm{i}(X+Y)}$. A general Hermitian H is a sum of many Pauli strings, and most pairs of them anticommute. So the naive product of circuits computes the wrong unitary. The error does not shrink as the circuit gets longer.

Theorem 3.3 (Trotter–Suzuki)**.** *Let $A_1, \ldots, A_m$ be Hermitian. Then*

$$\left(\mathrm{e}^{\mathrm{i}A_1/N}\mathrm{e}^{\mathrm{i}A_2/N}\cdots\mathrm{e}^{\mathrm{i}A_m/N}\right)^N \longrightarrow \mathrm{e}^{\mathrm{i}(A_1+\cdots+A_m)} \qquad \textit{as } N\to\infty,$$

with error $\mathcal{O}(1/N)$. An accuracy of δ therefore needs $N = \mathcal{O}(1/\delta)$ repetitions.

Proof sketch. Apply Proposition 3.2 with $A \to \mathrm{i}A_j/N$. One pass through the product differs from $\mathrm{e}^{\mathrm{i}(\sum_j A_j)/N}$ by the commutator terms. Each term carries two factors of $1/N$, so each is $\mathcal{O}(1/N^2)$. Repeating N times collects N such errors, and errors add by Remark 3.5. The total is $N\cdot\mathcal{O}(1/N^2) = \mathcal{O}(1/N)$. □

Remark 3.4 (the strong form of the statement)**.** Let us see why Theorem 3.3 says anything at all. For large N the product $\mathrm{e}^{\mathrm{i}A_1/N}\cdots\mathrm{e}^{\mathrm{i}A_m/N}$ is close to the identity. So is $\mathrm{e}^{\mathrm{i}(\sum A_j)/N}$. An approximation between two things that are both close to the identity has no value.

The content is that the approximation *survives being raised to the Nth power.* That is exactly the regime in which errors can pile up. They do pile up, but only linearly.

Remark 3.5 (cut the slices thinner, not fewer)**.** The fact that we used in the proof gets its own remark, because we use it all the time. If $\|U_j - V_j\| \le \epsilon_j$ for each j, then

$$\|U_m\cdots U_1 - V_m\cdots V_1\| \le \sum_{j=1}^{m}\epsilon_j.$$

Errors in a circuit add. They do not multiply. So to build a circuit of m gates to a total accuracy of ϵ, build each gate to accuracy ϵ/m.

3.3 The Universality Theorem

Definition 3.6 (universal gate set)**.** A set G of gates is *universal* when the following holds. For every n, every unitary U on n qubits and every $\delta > 0$, there is a finite circuit over G whose unitary $\tilde{U}$ satisfies $\left\| U - \tilde{U} \right\| \leq \delta$.

We make two remarks on the definition. First, when G is finite, we cannot avoid the approximation. Finitely many gates generate at most a countable set of circuits, and the unitary group is uncountable. Second, nothing in the definition says that the circuit is short. We treat that question in Section 3.5.

Theorem 3.7 (universality)**.** *The set* $\{R_x(\theta) : \theta \in \mathbb{R}\} \cup \{H, S, \textsc{cnot}\}$ *is universal.*

Proof. The proof has four steps, and we already proved three of them.

Step 1. By Theorem 2.1, write $U = \mathrm{e}^{\mathrm{i}H}$ with H Hermitian.

Step 2. By Theorem 1.5, expand $H = \sum_j J_j P_j$ over the 4^n Pauli strings P_j. Every coefficient J_j is *real*, because H is Hermitian.

Step 3. By Proposition 3.1 and the conjugation that follows it, each $\mathrm{e}^{\mathrm{i}J_j P_j}$ has an exact circuit over the gate set. Take $\theta = -2J_j$ in Figure 13.

Step 4. The terms $J_j P_j$ do not commute, so Theorem 3.3 gives

$$U = \mathrm{e}^{\mathrm{i}\sum_j J_j P_j} = \lim_{N \to \infty} \Big(\prod_j \mathrm{e}^{\mathrm{i}J_j P_j / N} \Big)^N .$$

Take $N = \mathcal{O}(1/\delta)$ and you get a circuit within δ of U. Each factor is a circuit by Step 3, so the whole product is a circuit over the gate set. □

Reversing the proof turns it to a recipe. To build a given unitary, take its logarithm to get a Hermitian. Expand that Hermitian in Paulis. Turn each Pauli term into a ladder-and-rotation circuit. Then interleave thin slices of all of them, many times over.
Also, never do this. The expansion in Step 2 has up to 4^n terms, and Step 4 repeats all of them $\mathcal{O}(1/\delta)$ times. The theorem proves that a circuit exists. It is not a compiler. It plays the same role as the lookup table that proves NAND gates universal for classical computation. That would be a pretty terrible way to build a program.

Remark 3.8 (why this proof and not the other one)**.** Nielsen and Chuang prove universality by a different route. Every unitary is a product of *two-level* unitaries, which act non-trivially only on a two-dimensional subspace spanned by two basis states. Each two-level unitary is a multiply-controlled single-qubit gate reached by a Gray-code walk. Each of those then splits into CNOT gates and single-qubit gates by Theorem 2.8.

That route gives an *exact* decomposition with the explicit count $\mathcal{O}(n^2 4^n)$. Read it if you want to know how a compiler works.

The route above fits this course, because we proved every ingredient in Lecture 4. It also generalises in a direction that we care about (see Remark 3.9).

Remark 3.9 (the same theorem, read as physics)**.** Step 2 wrote $H = \sum_j J_j P_j$. A physicist writes the same expression for the *Hamiltonian* of a system of interacting spin-$\frac{1}{2}$ particles. Here P_j is an interaction term and J_j is its coupling strength. The time evolution of that system for a time t is the unitary $\mathrm{e}^{-\mathrm{i}Ht}$, and the circuit of Step 4 simulates it.

So the universality proof is also an algorithm for simulating quantum systems. Writing down the state of n spin-$\frac{1}{2}$ particles on a classical computer takes 2^n numbers. On a quantum

computer it takes n qubits. The evolution then takes a number of gates polynomial in n and in the number of Hamiltonian terms. This is the oldest proposed application of quantum computers, and it is still the most plausible one. We proved it here in passing.

3.4 Other Universal Gate Sets

Theorem 3.7 uses a *continuum* of gates, because θ ranges over the reals. Real hardware cannot offer that. A rotation by θ and a rotation by θ plus a tiny amount are both valid gates, and no calibration tells them apart. The gate sets that matter in practice are finite.

Gate set	**Comment**
$\{R_x(\theta)\} \cup \{H, S, \text{CNOT}\}$	Theorem 3.7. Continuous, so it cannot be implemented directly in a fault-tolerant way.
Clifford + T	$\{H, S, \text{CNOT}, T\}$. *Finite.* The standard gate set of fault-tolerant quantum computing.
Clifford + any non-Clifford	The general statement. The Cliffords alone are not universal, as Section 3.6 shows, and adding almost anything outside them repairs that.
Toffoli + H	Surprising, because Toffoli alone is only classical. Two gates, one of them a permutation matrix, suffice for all of quantum computation.
Any entangler + all single-qubit gates	Any two-qubit gate that can create entanglement works in place of CNOT.

Remark 3.10 (why Clifford plus T, and what a T costs)**.** Error correction is the reason this set became standard, and it changes what cost means. On an error-corrected machine, Clifford gates can be performed in a way that resists noise and is comparatively cheap. The T gate cannot. Each one is produced by a procedure called magic state distillation. That procedure can consume hundreds or thousands of physical qubits and many rounds of error correction.

So for fault-tolerant algorithms the dominant cost is not the gate count, and not even the CNOT count. It is the *T-count*. A Toffoli, for example, costs 7 T gates. In Lecture 9, our resource estimates for factoring are T-counts.

Theorem 3.11 (Solovay–Kitaev, stated without proof)**.** *Let G be a finite universal gate set that is closed under inverses. Any single-qubit unitary can be approximated to accuracy ϵ by a circuit of $\mathcal{O}\big(\log^c(1/\epsilon)\big)$ gates from G, where c is a small constant. An efficient classical algorithm finds the circuit.*

In practice this means that the choice among finite universal gate sets does not matter. Any one of them simulates any other with only polylogarithmic overhead. In theory it means that accuracy ϵ is *exponentially cheaper* than a naive count suggests. You pay $\log(1/\epsilon)$ and not $1/\epsilon$. We use this in Lecture 7, Section 3.8, where we observe that a small rotation is expensive rather than free.

3.5 Universality Is Not Efficiency

Proposition 3.12 (the counting bound)**.** *Fix a finite gate set G of one-qubit and two-qubit gates. Then almost every n-qubit unitary needs circuits of length $\Omega\big(4^n/n\big)$ to approximate it to any fixed accuracy.*

Proof sketch. Count both sides. A circuit of length g is specified by choosing, g times, a gate from G together with the one or two wires it acts on. That gives at most $\big(|G|\, n^2\big)^g$ circuits.

The group of n-qubit unitaries has real dimension 4^n. Covering it with balls of radius ϵ needs at least $(c/\epsilon)^{4^n}$ of them. For the circuits to reach every unitary we need

$$\big(|G|n^2\big)^g \;\geq\; (c/\epsilon)^{4^n}, \qquad \text{so} \qquad g \;\geq\; \frac{4^n \log(c/\epsilon)}{\log(|G|n^2)} \;=\; \Omega\!\left(\frac{4^n}{\log n}\right). \qquad \square$$

Universality and efficiency are different claims, and the counting bound shows how far apart they are. Every unitary can be reached. The large majority can be reached only by circuits of exponential length, so in practice they cannot be reached at all.
The interesting question is therefore never "can a quantum computer do this?" It is "is the circuit short?" In Lectures 7 to 9 we look at a few specific unitaries whose circuits happen to be short and happen to be useful.

3.6 Gottesman–Knill

In Block 1 we promised to complicate the claim that entanglement is what makes a quantum computer fast. Here is the complication.

Theorem 3.13 (Gottesman–Knill, stated without proof)**.** *Take a circuit built entirely from Clifford gates, applied to a computational basis state and followed by measurements in the computational basis. A classical computer can simulate it in time polynomial in the number of qubits and the number of gates.*

The reason follows from the definition of a Clifford. We do not track the 2^n amplitudes of the state. Instead, we track the Pauli strings that stabilise it. For $|0\rangle^{\otimes n}$ these are $Z_1, \ldots, Z_n$, which is $\mathcal{O}(n^2)$ bits of data. In Section 2.3 we showed that each Clifford gate maps Pauli strings to Pauli strings, so a gate updates this data in $\mathcal{O}(n)$ steps. The state is never written down.

Consequences.

- The Bell circuit of Figure 5 is Clifford, so a classical computer can simulate it. The same holds for highly entangled states on thousands of qubits. *Entanglement is necessary for a quantum speed-up, but it is far from sufficient.*
- The Cliffords are therefore *not* a universal gate set. This is why Theorem 3.7 needed $R_x(\theta)$, and why the standard finite set is Clifford plus T.
- In the Clifford-plus-T picture, the T gates are where the quantum advantage lives. They are also the expensive ones. That is not a coincidence.

4 Block 4: Classical Computation Inside a Quantum Circuit

By the end of this block you can. . .

- Explain why a quantum circuit must be reversible, and what that forces the shape of U_f to be.
- Use the Toffoli gate to compute AND and NAND, and say why that makes it universal for classical computation.
- Prove the no-cloning theorem and say what it does and does not forbid.

- Build U_f from a classical circuit by compute, copy and uncompute, and show that skipping the uncomputation destroys interference.
- Convert a Boolean oracle into a phase oracle by phase kickback.
- State the principles of deferred measurement and implicit measurement.
- List the cost measures of a quantum circuit, and explain what the query model hides.

4.1 Reversibility, and the Shape of U_f

Every gate is unitary, so it is invertible, and therefore it is reversible. From the output you can always recover the input. Classical logic does not work this way. The gate AND takes two bits to one bit. From the output 0 you cannot tell which of three inputs produced it. The gate destroys information, and no unitary destroys information.

The fix is to keep the input. Take $f : \{0,1\}^n \to \{0,1\}^m$ and define

$$U_f \left|x\right\rangle \left|y\right\rangle = \left|x\right\rangle \left|y \oplus f(x)\right\rangle . \tag{6}$$

This is the *Boolean oracle* for f, also called the standard oracle. In Lecture 4, Exercise A.16, you proved that it is unitary for every f. The reason is that $\oplus$ is its own inverse, so $U_f^2 = I$ and U_f only permutes the computational basis.

Why not the obvious thing?

The natural-looking map $\left|x\right\rangle \mapsto \left|f(x)\right\rangle$ is *not* unitary unless f is a bijection. If $f(x) = f(x')$ for $x \neq x'$, the map sends two orthonormal basis vectors to the same vector, and no unitary does that. The $\oplus$ in (6) is not a matter of style. It is the cheapest way to fit an arbitrary function inside a reversible machine.

Next we build U_f from gates. One gate does the classical half of the job.

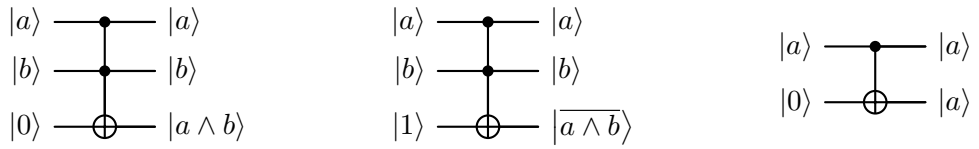


Figure 14: Classical logic in a quantum circuit. With the target preset to $\left|0\right\rangle$, a Toffoli computes AND. Preset to $\left|1\right\rangle$, it computes NAND. A CNOT with target $\left|0\right\rangle$ copies a bit. The gate NAND with fan-out is universal for Boolean logic. So any classical circuit of g gates becomes a reversible circuit of $\mathcal{O}(g)$ Toffoli and CNOT gates.

4.2 No-Cloning

The rightmost circuit in Figure 14 copies a bit. It is natural to ask whether the same trick copies a qubit. It does not.

Theorem 4.1 (no-cloning)**.** *There is no unitary U on two qubits with $U\big(\left|\psi\right\rangle \otimes \left|0\right\rangle\big) = \left|\psi\right\rangle \otimes \left|\psi\right\rangle$ for every state $\left|\psi\right\rangle$.*

Proof. Suppose such a U exists, and let $\left|\psi\right\rangle$ and $\left|\varphi\right\rangle$ be any two states. In Lecture 4, Block 3 we showed that unitaries preserve inner products, so

$$\langle\psi|\varphi\rangle = \big(\left|\psi\right\rangle \otimes \left|0\right\rangle\big)^\dagger \big(\left|\varphi\right\rangle \otimes \left|0\right\rangle\big) = \big(\left|\psi\right\rangle \otimes \left|\psi\right\rangle\big)^\dagger \big(\left|\varphi\right\rangle \otimes \left|\varphi\right\rangle\big) = \langle\psi|\varphi\rangle^2 ,$$

where we used $\langle 0|0\rangle = 1$ and the rule for the tensor inner product. So $z = z^2$ for $z = \langle\psi|\varphi\rangle$, which forces $z \in \{0, 1\}$. Any two states the machine clones are either identical or orthogonal. □

Specifics on no-cloning.

- It *does not* forbid copying states drawn from a known orthogonal set. A CNOT copies $|0\rangle$ and $|1\rangle$ perfectly, and that is all Section 4.3 needs.
- It *does not* forbid preparing many copies of a state you know how to prepare. Cloning means copying an *unknown* state with a *single* fixed circuit.
- It follows from *linearity*, not from measurement and not from the uncertainty principle. The proof above never mentions a measurement.
- It does forbid the move that you want to make in Section 4.3. You cannot save a copy of an intermediate quantum state and come back to it later.

4.3 Garbage, and How to Deal With It

Suppose that we compile a classical circuit for f into Toffoli and CNOT gates as in Figure 14. We do not get (6). We get something messier, because a real computation uses scratch variables:

$$U : \ |x\rangle\,|0\rangle\,|0\rangle \ \longmapsto \ |x\rangle\,|f(x)\rangle\,|g(x)\rangle\,, \tag{7}$$

where $g(x)$ is whatever the intermediate wires hold. We call it the *garbage*. On a classical computer, we erase it. On a quantum computer, we cannot. Erasure is the map $|0\rangle \mapsto |0\rangle$, $|1\rangle \mapsto |0\rangle$, which sends two orthogonal states to the same state and is therefore not unitary.

Garbage is not harmless

It is tempting to ignore the garbage register, because we never look at it. That is wrong, and we now look at the reason in detail.
Suppose an algorithm produces

$$\frac{1}{2^{n/2}} \sum_x (-1)^{f(x)}\,|x\rangle \otimes |g(x)\rangle$$

with a different garbage value $g(x)$ for each x. It then applies $H^{\otimes n}$ to the first register and measures. The amplitude of $|0\cdots0\rangle$ is now a *vector* in the garbage register, namely $2^{-n}\sum_x (-1)^{f(x)}\,|g(x)\rangle$. Its squared norm is

$$\frac{1}{4^n} \sum_x 1 = \frac{1}{2^n},$$

because the states $|g(x)\rangle$ are orthogonal and nothing cancels.
Compare the garbage-free case. There the amplitude is the *number* $2^{-n}\sum_x (-1)^{f(x)}$, which is ±1 for constant f and 0 for balanced f. The garbage register records which x each branch came from, and branches that can be told apart cannot interfere. If you omit the uncomputation, every speed-up in Lectures 7 to 9 dies at this step.

The solution uses the fact that a circuit can always run backwards. If $U = G_k \cdots G_1$ then $U^\dagger = G_1^\dagger \cdots G_k^\dagger$, and each $G_j^\dagger$ is a gate that we have. So we compute, then copy the answer out with CNOT gates, then run the computation backwards. The copy is legitimate. Theorem 4.1 forbids cloning, but here we copy only basis states.

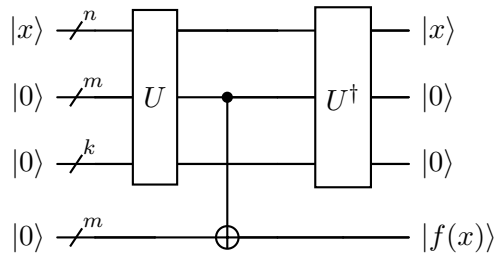


Figure 15: Compute, copy, uncompute. The three registers below the input hold the output, the garbage, and a clean copy of the output. First U writes $f(x)$ and $g(x)$. Then a layer of m CNOT gates copies $f(x)$ to the bottom register. Then $U^\dagger$ erases both $f(x)$ and $g(x)$ and restores the zeros. The net effect on the first and last registers is U_f of (6).

Theorem (Bennett), informally. If a classical circuit of g gates computes f, then a quantum circuit of $\mathcal{O}(g)$ gates implements U_f with no leftover garbage. So in Lectures 7 to 9 we can treat U_f as given. In Lecture 9, where the oracle is modular exponentiation, we cash this in and count the gates.

4.4 The Phase Oracle

Algorithms rarely want $f(x)$ written into a register. They want $f(x)$ written into a *phase*, because phases interfere and register contents do not. For $f : \{0,1\}^n \to \{0,1\}$ define the *phase oracle*

$$U_f^{\pm} |x\rangle = (-1)^{f(x)} |x\rangle . \tag{8}$$

We already built a small version of it in Section 2.5.

Proposition 4.2 (kickback, oracle version)**.** *We have* $U_f |x\rangle |-\rangle = (-1)^{f(x)} |x\rangle |-\rangle$*, and therefore*

$$U_f\Big(\sum_x c_x |x\rangle\Big) |-\rangle = \Big(\sum_x c_x (-1)^{f(x)} |x\rangle\Big) |-\rangle .$$

Proof. With a single output bit, (6) applies $X^{f(x)}$ to the target. Since $X |-\rangle = - |-\rangle$, the target returns unchanged and the state collects $(-1)^{f(x)}$. Linearity gives the superposition case. The target factors out untouched, so you can discard it. □

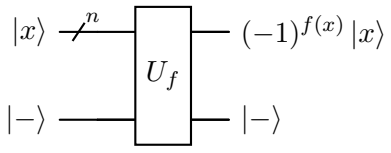


Figure 16: Phase kickback. The target register is prepared in $|-\rangle$, the oracle leaves it alone, and you can prepare it once and reuse it. Converting a Boolean oracle to a phase oracle costs one ancilla and two Hadamards.

The general form, on which we build Lecture 8. If $U |u\rangle = \mathrm{e}^{2\pi \mathrm{i} \varphi} |u\rangle$ then

$$\Lambda(U)\ |c\rangle |u\rangle = \mathrm{e}^{2\pi \mathrm{i} c\varphi} |c\rangle |u\rangle \qquad \text{and} \qquad \Lambda(U)\ |+\rangle |u\rangle = \frac{|0\rangle + \mathrm{e}^{2\pi \mathrm{i} \varphi} |1\rangle}{\sqrt{2}} \otimes |u\rangle .$$

The eigenvalue moves out of the target, where it was a global phase and cannot be read. It now sits on the control as a relative phase, where it can be measured. Phase estimation repeats this observation with U^{2^j} for several values of j, and then applies an inverse Fourier transform.

4.5 Measurement in Circuits

Measurement is the only part of the model that is not unitary. Two principles govern where you can draw it.

Deferred measurement. Take a measurement whose outcome only decides whether to apply a later gate. You can replace it by a quantum-controlled gate and move the measurement to the end of the circuit. Classical control is never *necessary*. It is only convenient.
Implicit measurement. Take a wire that is left unmeasured at the end of a circuit. You can assume that it was measured, because nothing that happens to it affects the statistics of the wires you do measure.

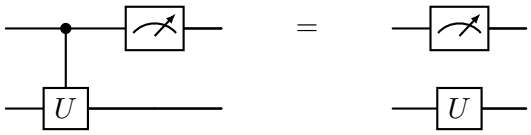


Figure 17: Deferred measurement. Measuring the control before or after a controlled gate gives the same statistics. So a classically controlled gate, on the right, and a quantum-controlled gate, on the left, are interchangeable.

Remark 4.3 (implicit measurement is why garbage kills)**.** The second principle gives the rigorous version of the warning in Section 4.3. Leaving the garbage register unmeasured at the end of the circuit is the same thing as measuring it. Measuring a register that records which branch the computation took is exactly what destroys the superposition over branches. Uncomputation is not a matter of tidiness. It is the difference between an algorithm and a random number generator.

A measurement in any basis is a basis change followed by a computational-basis measurement. If $V|j\rangle = |v_j\rangle$, then measuring in the basis $\{|v_j\rangle\}$ means applying $V^\dagger$ and then measuring. To measure X, apply H first. This is how the Bell circuit of Figure 5, run backwards, measures in the Bell basis.

4.6 Counting Cost (What is Hidden by the Oracle)

Measure	What it counts, and when it matters
gate count (size)	total gates over a stated elementary set. Always say which set.
depth	number of columns in the diagram. The closest thing to running time.
two-qubit count	CNOT gates. The dominant error source on today's hardware.
T-count	non-Clifford gates. The dominant cost on an error-corrected machine.
width	number of qubits, including ancillas.
query count	calls to U_f, charging nothing for anything else.

The query model hides the oracle

In Lectures 7 and 8 we prove separations in *query* complexity. The quantum algorithm calls U_f once where a classical one needs exponentially many calls. The statement is real and provable, but it charges nothing for building U_f. Two caveats follow.

First, a query separation is not automatically a running-time separation. The oracle can be expensive, or it can fail to exist. Second, the exponential separation of Deutsch–Jozsa holds only against *deterministic* classical algorithms. Students rarely hear this. A classical algorithm that is allowed a small error probability solves the same problem with a few random queries. Simon's problem gives the first honest exponential separation. Shor's algorithm is impressive because its oracle, modular exponentiation, is something we can build.

4.7 Superstitions about Superposition

The misconception that we kill in the next lecture

Here is a tempting story. A register of n qubits holds 2^n amplitudes. Prepare $H^{\otimes n}|0\rangle^{\otimes n}$ and apply U_f, and the machine evaluates f on all 2^n inputs at once:

$$U_f \frac{1}{2^{n/2}} \sum_x |x\rangle |0\rangle = \frac{1}{2^{n/2}} \sum_x |x\rangle |f(x)\rangle .$$

Every clause of that is true. The story is still false, and the circuit shows why. *Measure it.* The outcome is a single pair $\big(x, f(x)\big)$ with x chosen uniformly at random. That is what you get by picking x at random on a classical computer and evaluating f once. You gain nothing. A measurement returns n bits, not 2^n amplitudes, and no amount of superposition changes that.

A quantum algorithm instead arranges for the 2^n amplitudes to *interfere*. The one string that you can read is then a string that you want. The oracle writes a global property of f into the signs of the amplitudes. A final layer of gates, a Fourier transform, converts those signs into a measurement outcome. We open Lecture 7 with this circuit plus one extra Hadamard layer. That extra layer is the whole difference between a random number generator and an algorithm.