CPEN 400Q Lecture 11
Quantum compilation I

Monday 10 February 2025

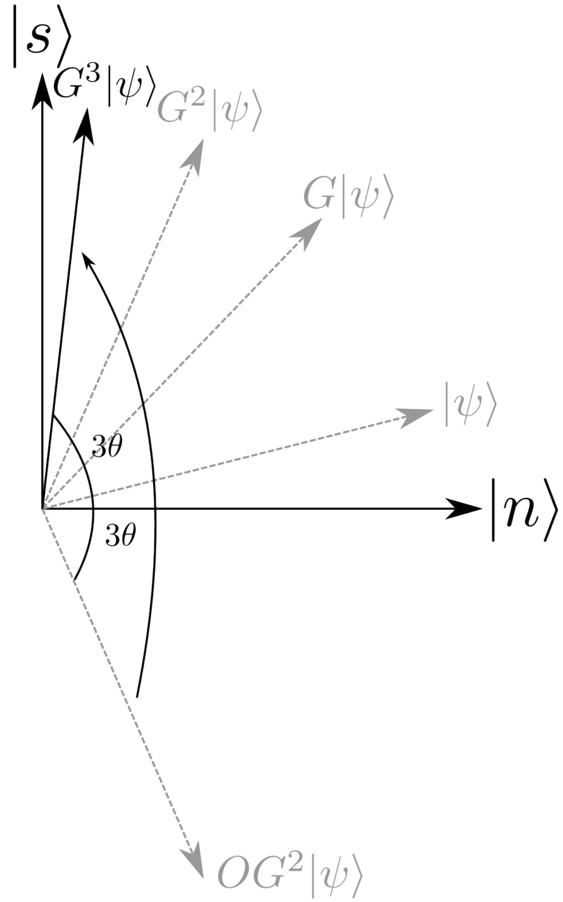
Announcements

- Quiz 4 today
- Literacy Assignment 1 due Wednesday 23:59
- Assignment 2 due Thursday 27 Feb at 23:59
- Project details available soon on PrairieLearn
- Tutorial tomorrow (hands-on assignment about compilation!)

Quiz

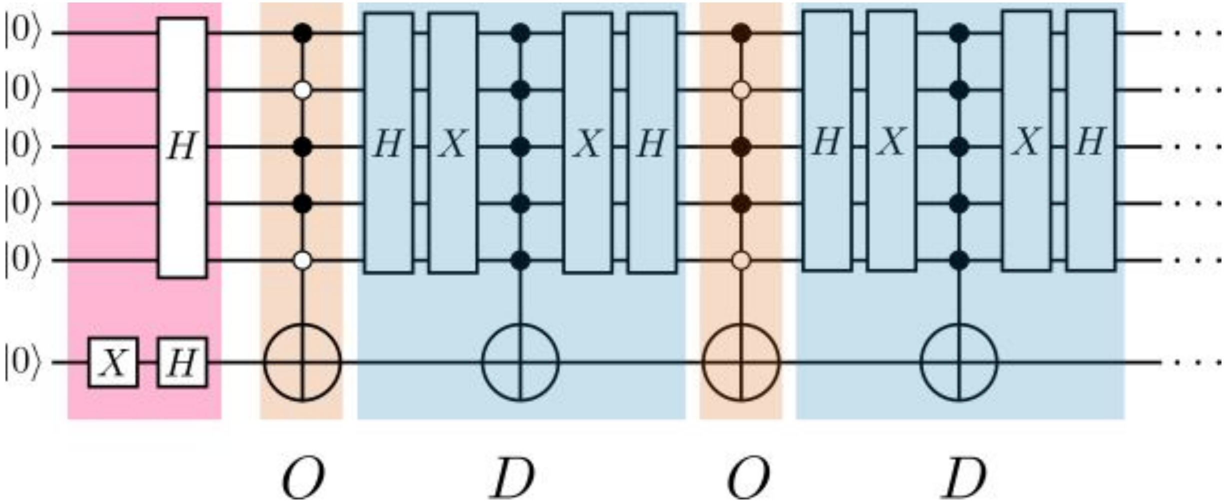
Last time

We discussed how Grover search of an unstructured space has a square-root speedup in **query complexity** over classical methods.



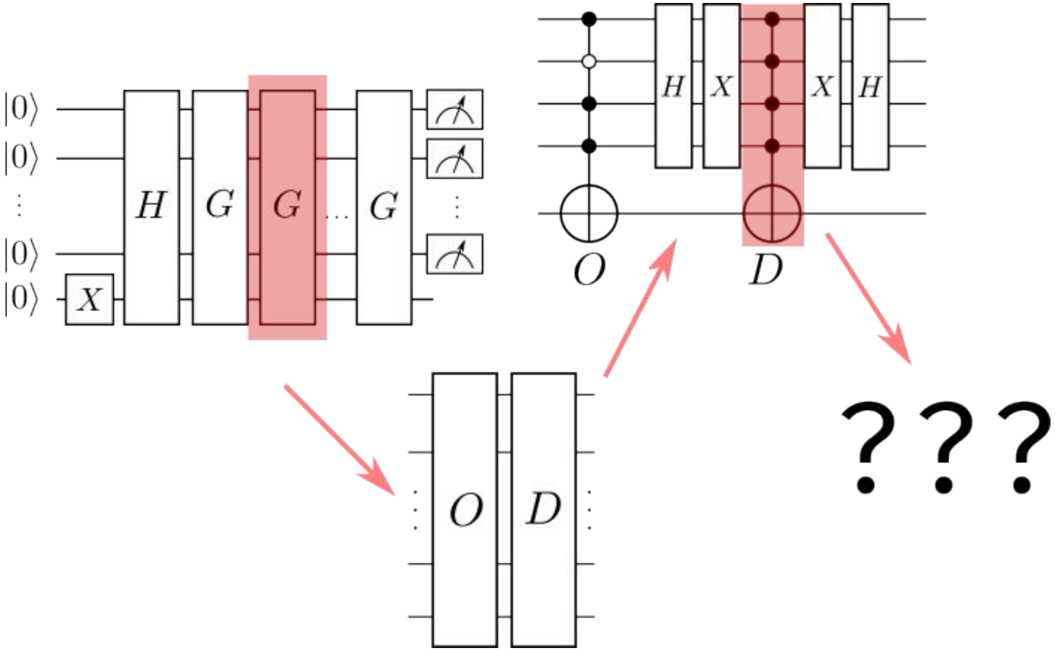
Last time

We implemented Grover's algorithm in PennyLane



Last time

We finished off with this picture



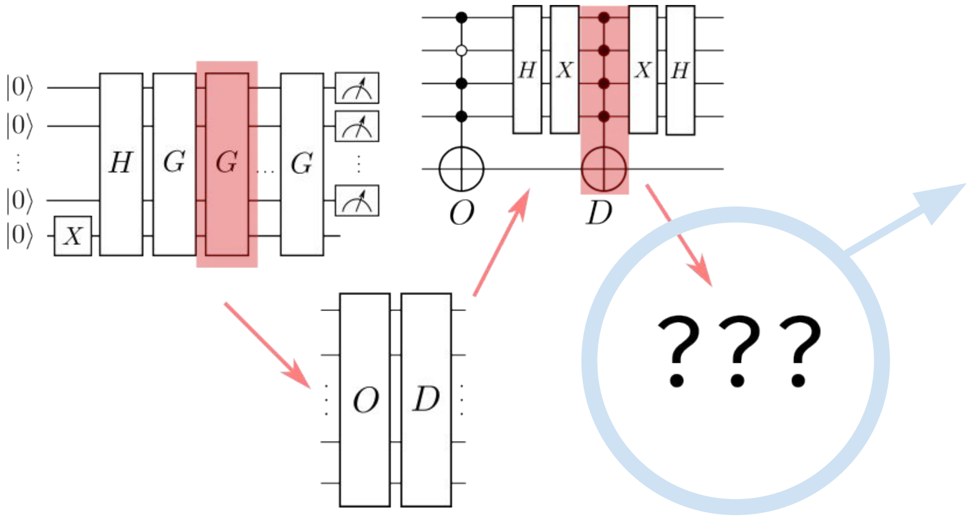
Today

Learning outcomes:

- explain the limitations of query complexity as a metric
- define and calculate **cost metrics** of quantum circuits
- define and list key **universal** gate sets
- perform local **optimizations** on quantum circuits
- **decompose** multi-controlled operations

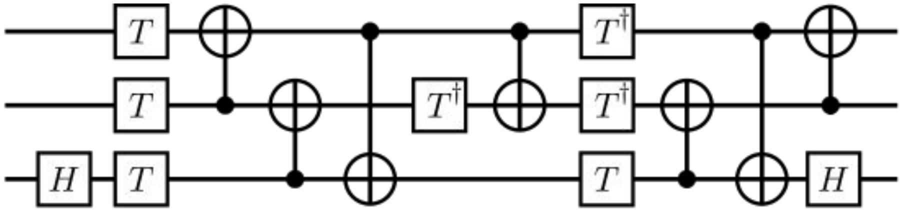
In tomorrow’s tutorial assignment and Wednesday’s lecture, we will put this into context within the 💗**quantum compilation stack**💗

Quantum circuit synthesis and optimization

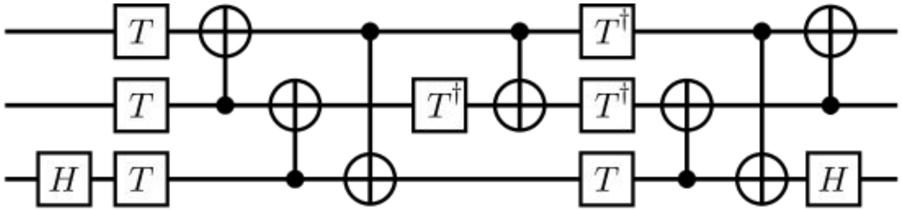


- How do we decide what goes here?
- What criteria determine whether our choice is “good”?
- How do we improve on our choices?

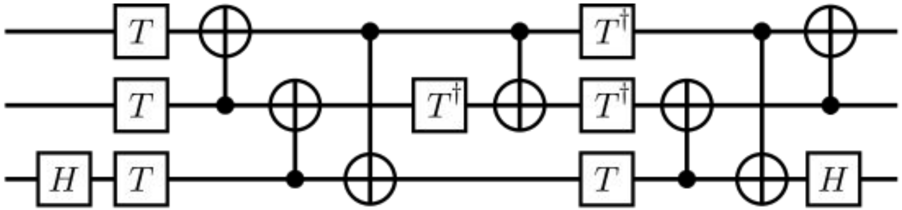
Circuit width



Gate count

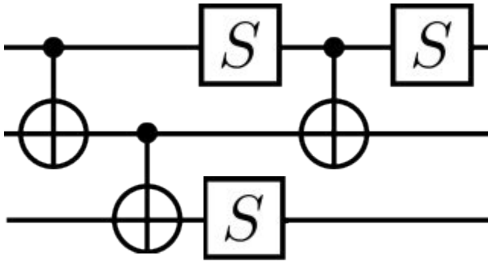


Circuit depth



Circuit depth

The circuit depth is the length of the longest path in its **directed acyclic graph (DAG)** representation.

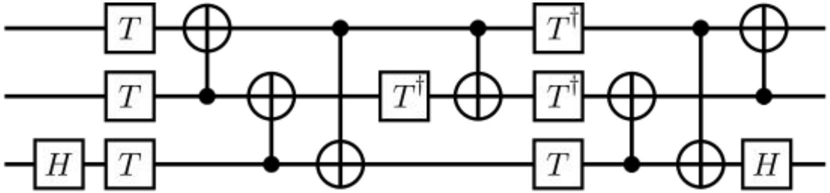


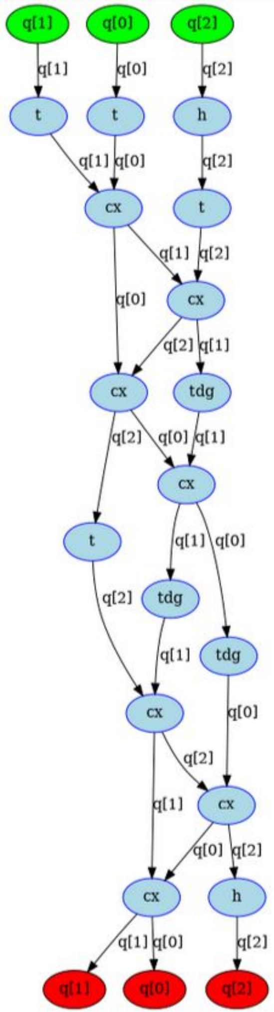
Key metrics

Circuit depth

Generally don’t do by hand; NetworkX is your best friend.

https://networkx.org/

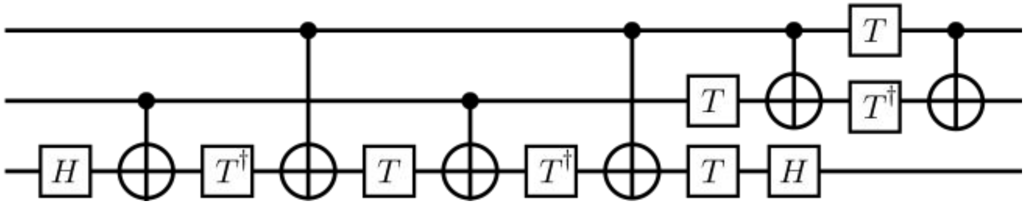




(DAG generated in Qiskit)

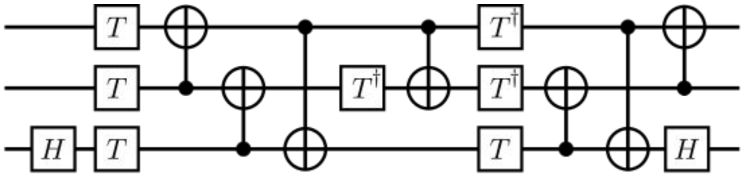
Exercise

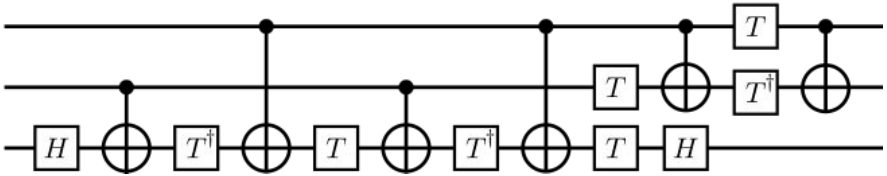
Determine resource counts of the following circuit.



Circuit synthesis and decomposition

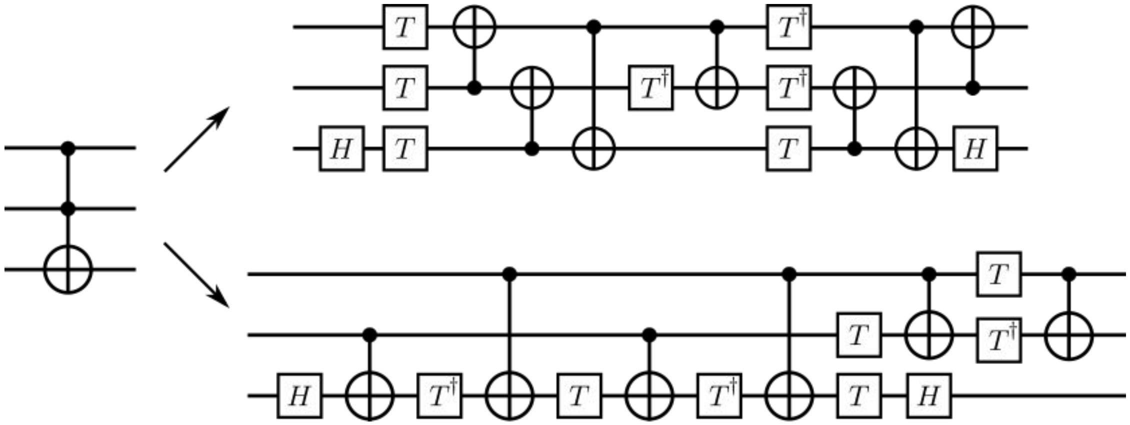
What do these circuits do?





Circuit synthesis and decomposition

What do these circuits do?



Circuit synthesis

Given:

- arbitrary unitary matrix U
- a finite set of gates $\{G_k\}$
- a target precision, ϵ

Find

$$U' = C_1 C_2 \cdots C_m$$

s.t.

$$||U - U'|| < \epsilon, C_i \in \{G_k\}$$

Universal gate sets

A $\{G_k\}$ for which this can be done is called a **universal gate set.**

Examples for a single qubit:

- Any two Pauli rotations (exact)
- Hadamard and T (approximate)

Theoretical bound: $O(\log^c(1/\epsilon))$ gates, c varies from 1-4 depending on assumptions.

Universal gate sets

Any single-qubit unitary can be expressed using a sequence of 3 Pauli rotations around a minimum of 2 unique axes

- ZYZ
- XYX
- XYZ
- 9 other combinations *(Euler angles)*

qml.Rot

class Rot(*phi, theta, omega, wires, id=None*)

Bases: `pennylane.operation.Operation`

Arbitrary single qubit rotation

$$R(\phi,\theta,\omega) = RZ(\omega)RY(\theta)RZ(\phi) = \begin{bmatrix} e^{-i(\phi+\omega)/2}\cos(\theta/2) & -e^{i(\phi-\omega)/2}\sin(\theta/2) \\ e^{-i(\phi-\omega)/2}\sin(\theta/2) & e^{i(\phi+\omega)/2}\cos(\theta/2) \end{bmatrix}.$$

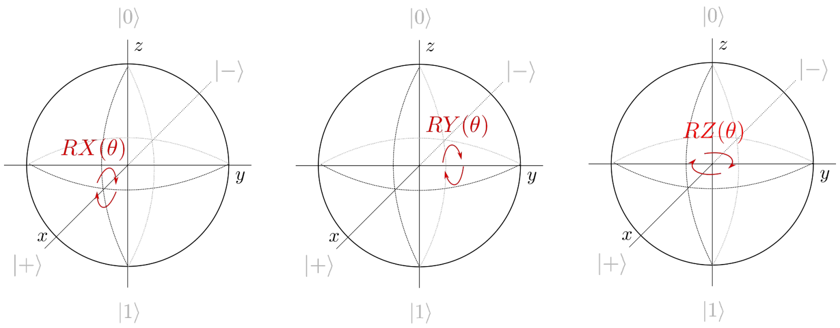


Image: Xanadu Quantum Codebook Node I.6

Universal gate sets

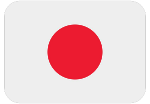
Using a different gate set is like *speaking a different language*.

I would like to use a real quantum computer.

I would like to use a real quantum computer.

J’aimerais utiliser un vrai ordinateur quantique.

I would like to use a real computer that is quantum.

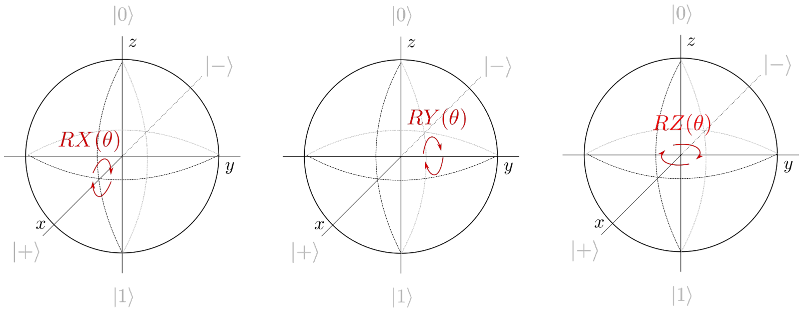


本当の量子コンピュータを作りたい。

A real quantum computer, I would like to use.

Hardware-specific example

RX/*RY*/*RZ* are **arbitrary-angle** rotations about **fixed axes**



IonQ's trapped ion hardware native gates: *GPI*/*GPI2*, **fixed-angle** rotations about **arbitrary axes** in *xy*-plane

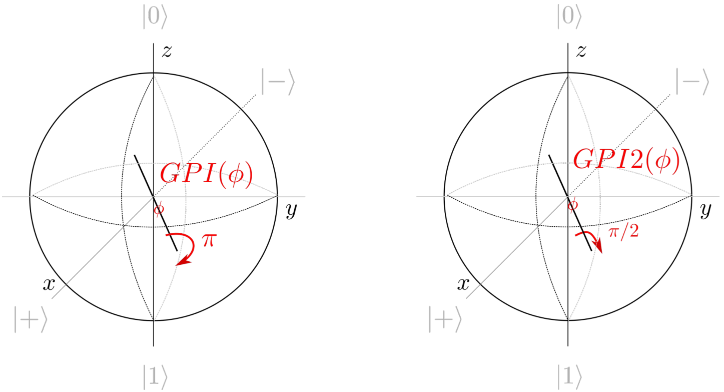


Image adapted from Xanadu Quantum Codebook Node I.6
See blog post: https://pennylane.ai/blog/2023/06/the-ionizer-building-a-hardware-specific-transpiler-using-pennylane

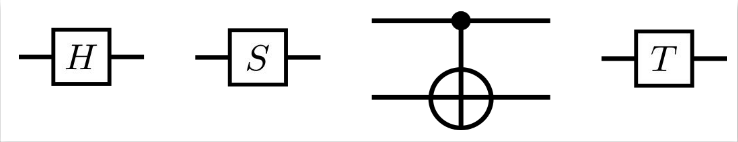
Exercise

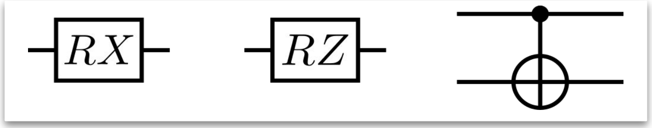
Express Pauli X in terms of H and T.

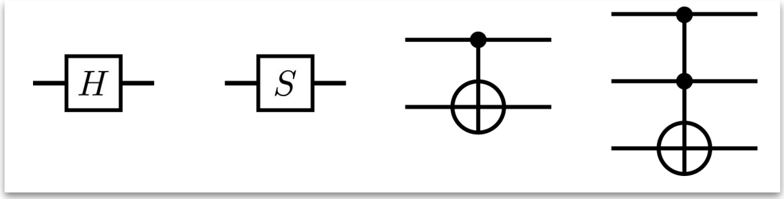
Hint: start by expressing it using H and Z.

Universal gate sets

Multi-qubit gate sets

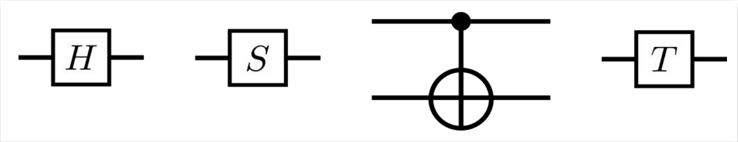






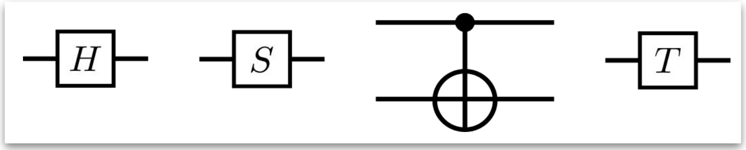
Exercise

Express CZ using gates from



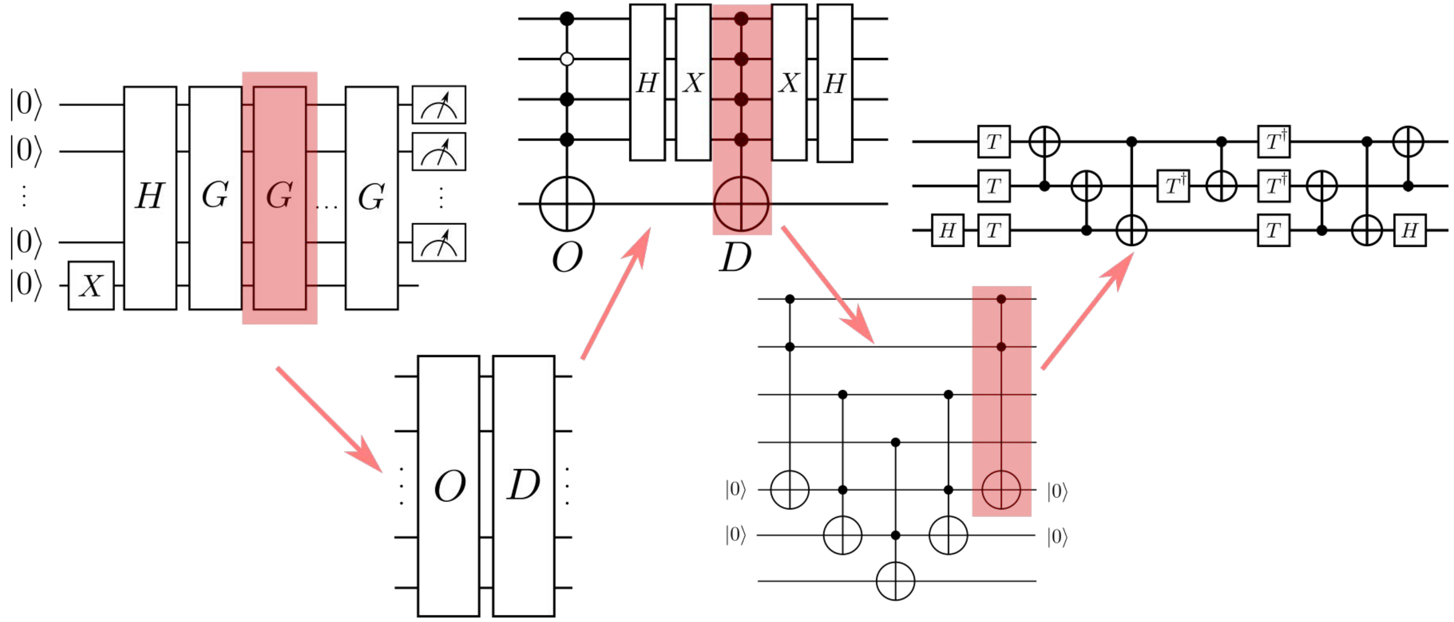
Take-home exercise

Express controlled Hadamard using gates from



Circuit decomposition

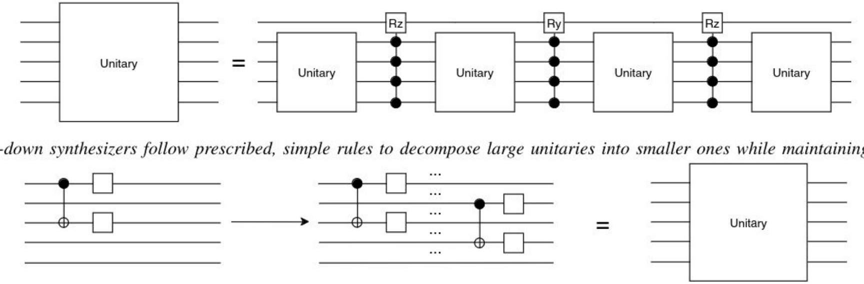
Decompositions of many common operations are **known**.



Synthesis and decomposition techniques

For under-specified or arbitrary operations, or for special gate sets, we may need to ***find*** a decomposition.

- search problem
- number-theoretic techniques
- numerical methods



(a) *Top-down synthesizers follow prescribed, simple rules to decompose large unitaries into smaller ones while maintaining equality.*

(b) *Bottom-up synthesizers start with an empty circuit and build up to equality.*

Figure 1: *Quantum synthesizers are either top-down or bottom-up.*

Image: E. Younis, K. Sen, K. Yelick, and C. Iancu. *QFAST: Conflating Search and Numerical Optimization for Scalable Quantum Circuit Synthesis.* arXiv:2103.07093 [quant-ph]

Challenges

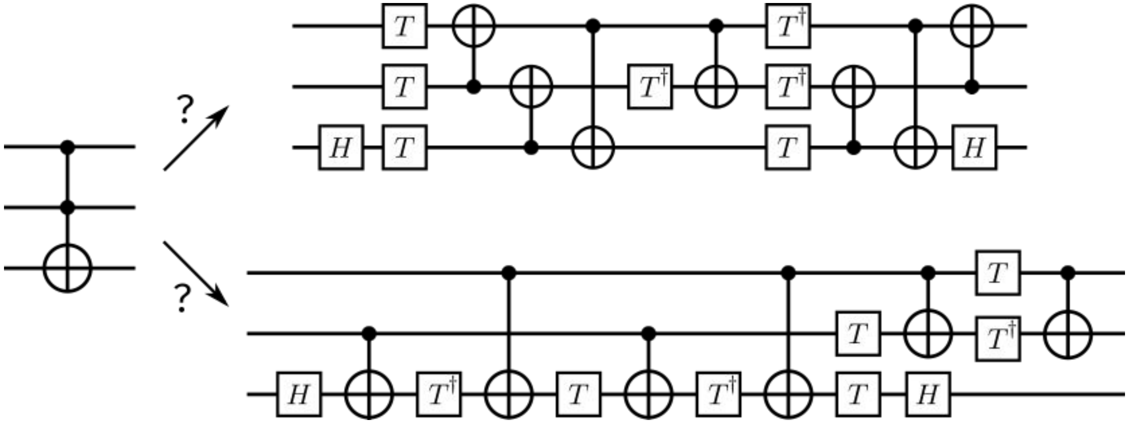
Synthesis is hard!

- algorithm complexity and computational resources

Challenges

Synthesis is hard!

- algorithm complexity and computational resources
- how to choose the best decomposition to *enable further optimization* down the line?



Challenges

Synthesis is hard!

- algorithm complexity and computational resources
- how to choose the best decomposition to *enable further optimization* down the line?
- manage tradeoffs between various resources

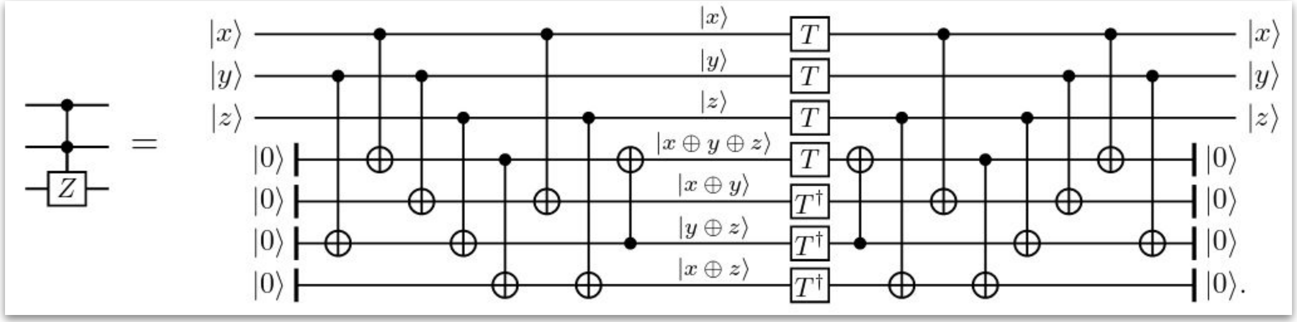
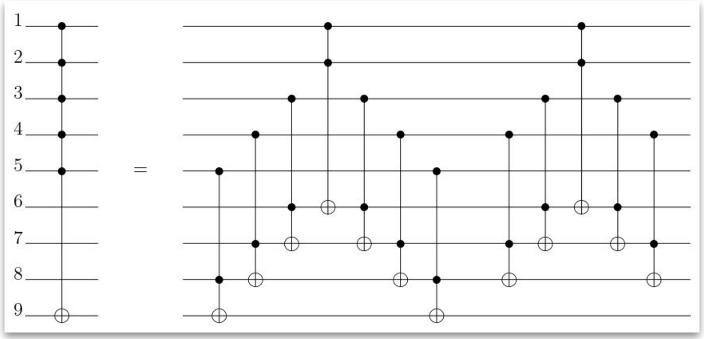


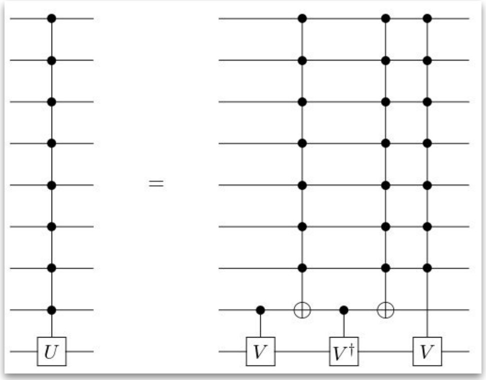
Image: P. Selinger (2013) *Quantum circuits of T -depth one*. Phys. Rev. A 87, 042302 (2013)

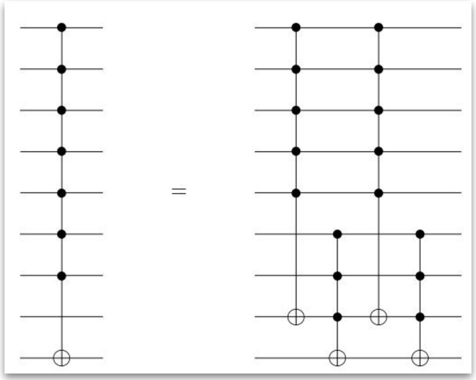
Example

Let’s decompose a multi-controlled NOT.

Multi-controlled NOT decomposition strategies



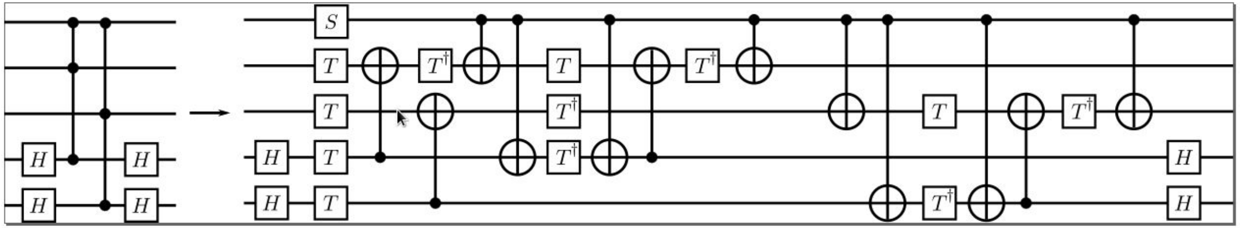




Circuit optimization

Machine-independent (or dependent) sequence of *passes* to reduce:

- gate count (overall, or for particular gate)
- circuit depth
- qubit count



Optimization passes

Passes have varying levels of complexity:

- inverse cancellation

Optimization passes

Passes have varying levels of complexity:

- inverse cancellation
- rotation merging

Optimization passes

Passes have varying levels of complexity:

- inverse cancellation
- rotation merging
- template matching

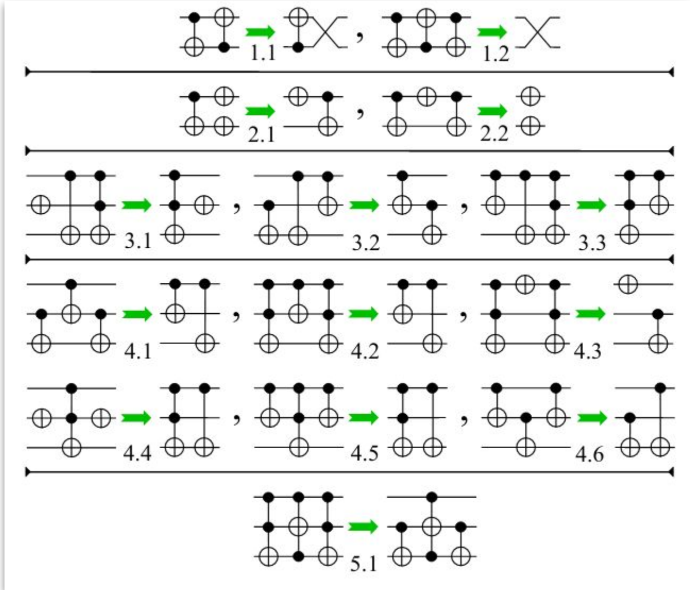
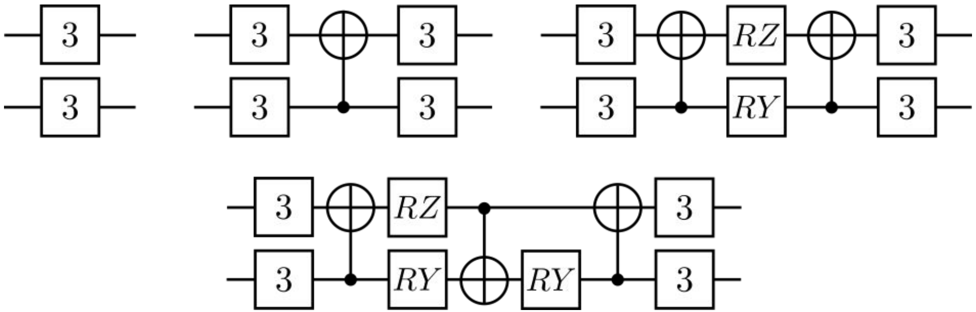


Image: D. M. Miller, D. Maslov, G. W. Dueck. *A Transformation Based Algorithm for Reversible Logic synthesis*. DAC 2003.

Optimization passes

Passes have varying levels of complexity:

- inverse cancellation
- rotation merging
- template matching
- two-qubit block *resynthesis*

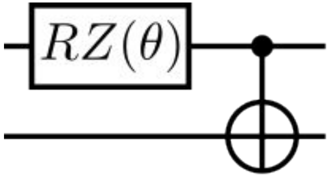


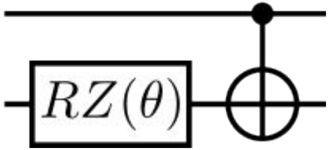
Reference: V. V. Shende, I. L. Markov, and S. S. Bullock (2004) *Minimal universal two-qubit controlled-NOT-based circuits*. Phys. Rev. A 69 062321.

Commutation

Commutation relations can help us move gates “through” each other to enable optimizations.

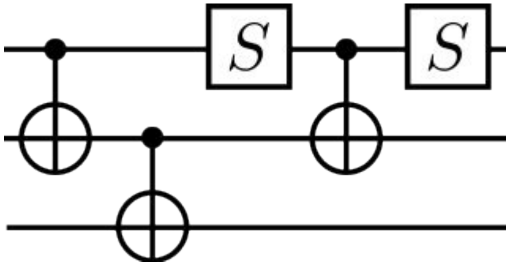
Exercise: do the following gates commute?





Exercise

Determine this circuit’s resource counts, then optimize it.
What is the minimum depth?



Next time

Content:

- The quantum compilation stack (from top to bottom)

Action items:

- A2 and Lit A1 (last question from A2 will be posted this week)
- Tutorial Assignment 2 tomorrow

Recommended reading:

- Nielsen and Chuang 4.5
- Transforms blog post (*syntax is out-of-date, but idea holds*)
- Explore the **qml.transforms** module