CMPT 478/981 Spring 2025
Quantum Circuits & Compilation

Matt Amy

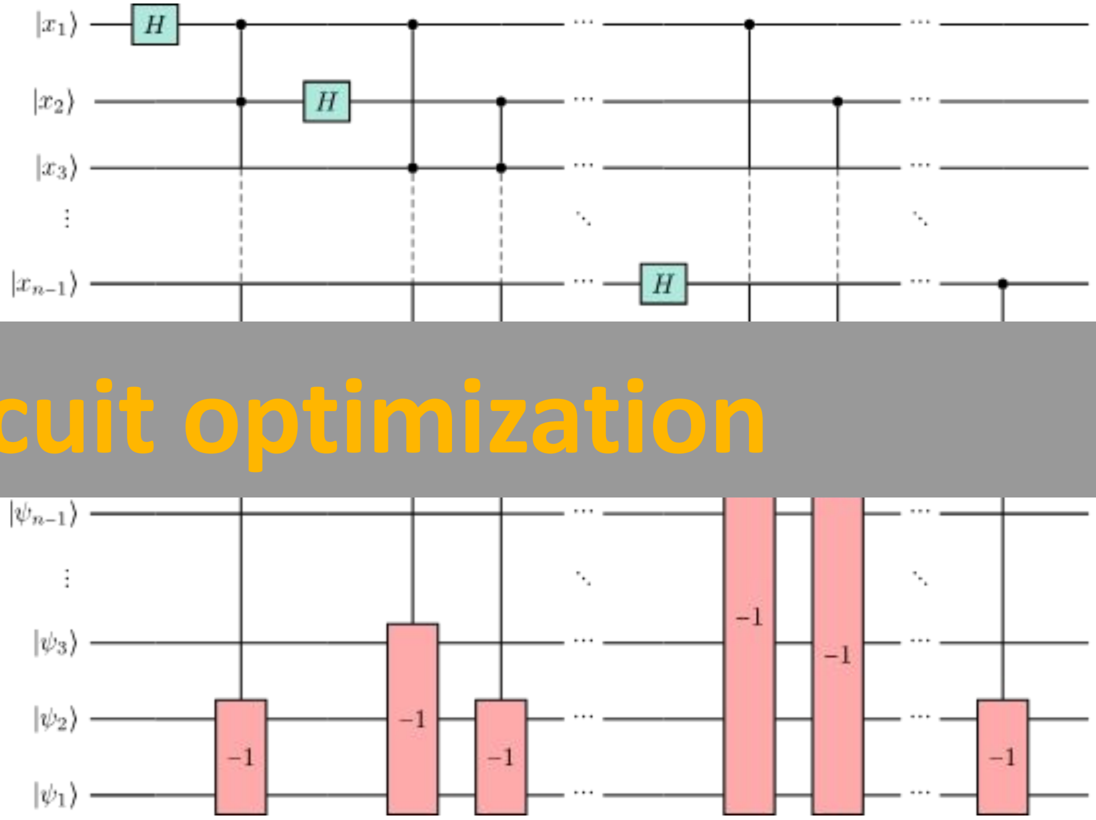
Today’s agenda

- Quantum circuit optimization
- Equational/re-writing theories
- Representations for optimization
- Housekeeping
    - Assignment due today
    - Decide on a project idea if you haven’t already
    - Paper presentations

Paper presentations

- Last **two weeks** of class (March 27th & April 3rd)
- **Everyone enrolled** will give 1 paper presentation
- Presentations will be 30 minutes
  - 20-ish minutes presentation
  - 10-ish minutes for questions/discussion
- I’ll post a list of possible papers, but you can choose any (in-scope) paper that interests you

Circuit optimization



The circuit optimization problem

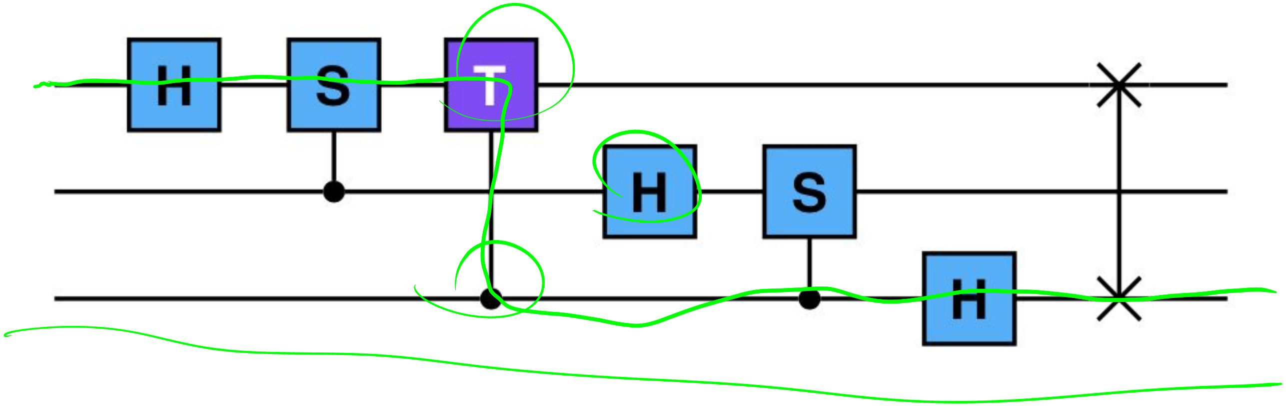
Given some circuit C over a gate set G and cost model c: Circuits(G) → R, find some equivalent (as partial isometries) circuit C' with c(C') < c(C)

Cost models:

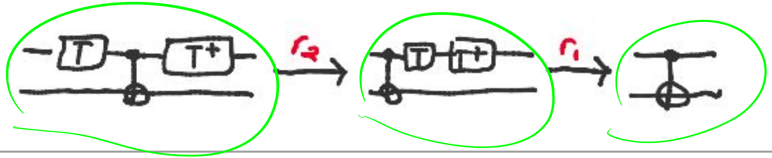
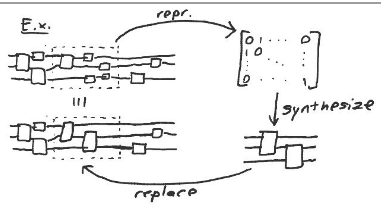
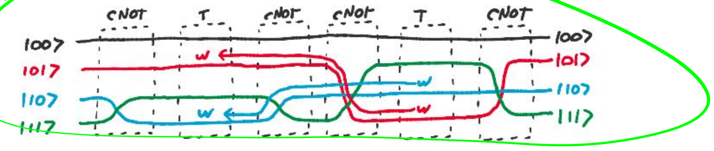
- T-count (dominant factor in surface code volume)
- CNOT-count (dominant factor in hardware fidelity)
- Total gate count (more relevant as T-state distillation gets cheaper)
- Depth (dominant factor in hardware compute time)
- T-depth (dominant factor in FT compute time with certain assumptions)

A word on depth

- Depth = length of a critical path
- Simple computation! (so don't mess it up or complicate it)
    - Step through the circuit & update length of outgoing critical paths



Approaches to circuit optimization

Quantum	Classical analogue
Re-writing based  	Peephole-optimization
Re-synthesis based  	Dynamic re-compilation?
Property/analysis based  	Dataflow optimization, Loop optimization, All other compiler optimizations

Re-writing

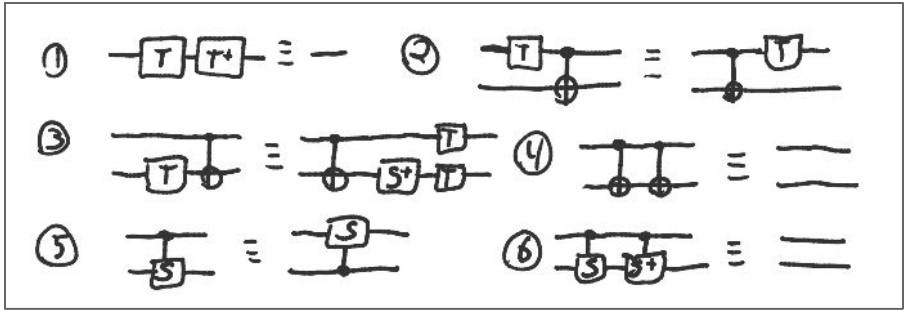
$|x_1\rangle$
$|x_2\rangle$
$|x_3\rangle$
$|x_{n-1}\rangle$
$|\psi_{n-1}\rangle$
$|\psi_3\rangle$
$|\psi_2\rangle$
$|\psi_1\rangle$
H
H
H
−1
−1
−1
−1
−1
−1

Re-writing

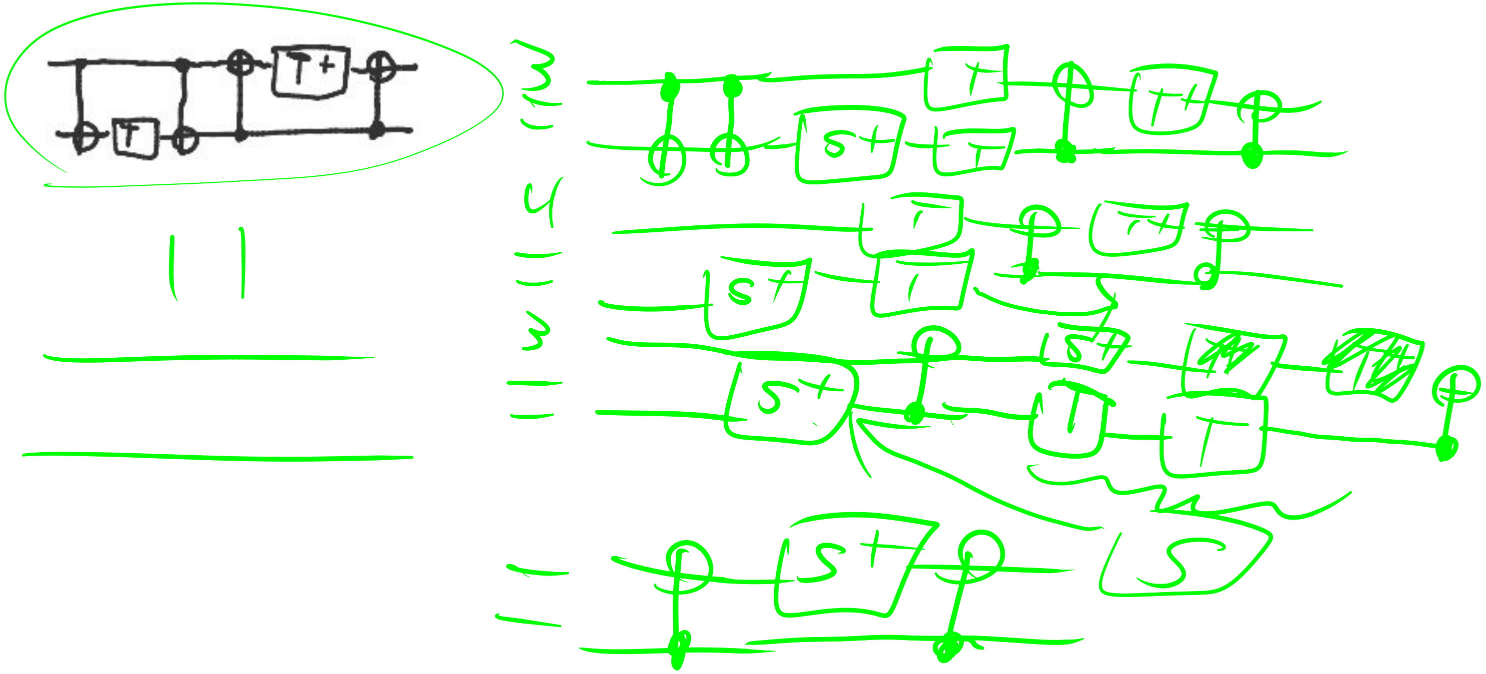
- Basic strategy: given a database *D* of re-write rules {c → c'}
    - Scan for a match with the LHS of some rule, replace with RHS
    - Repeat until no re-writes possible
    - Complexity? $O(|c|^2 |D| \max_{|c| \in D} |c|)$
- Effective when not much useful structure (e.g. hardware ansatz)
- Considerations in designing D
    - Confluence (~~does the order matter~~?)
    - Cost non-increasing (does every rewrite produce a strictly better circuit?)
    - Terminating (does the generic strategy terminate?)
    - Completeness (is every equivalent circuit reachable?)

For reasonable performance, typically need cost **increasing** rules

Example



Optimize:



(Formal) equational theories

A rewriting system drops these

- Given a gate set G, an equational theory of circuits over G is the equivalence closure (reflexive, symmetric, transitive closure) of a relation on G-circuits,

$$R \subseteq \langle G \rangle \times \langle G \rangle$$

ev(C) = matrix of C

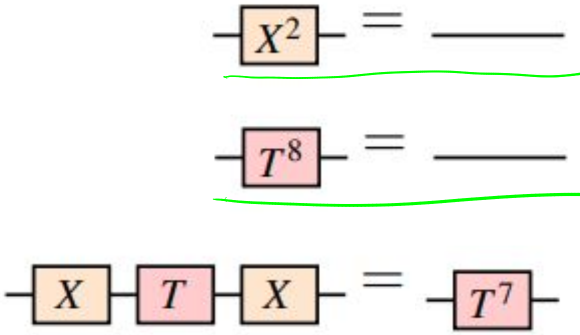
- The equational theory is:
  - **Sound** if $C' \in [C]_R \implies ev(C') = ev(C)$
  - **Complete** if $ev(C') = ev(C) \implies C' \in [C]_R$
- A sound & complete equational theory gives a presentation (and vice versa)
- Typically prove completeness by giving (non-optimal) normal forms

Example: dihedral group

- Circuits over <X, T> are (up to global phase) a linear representation of the Dihedral group of order 8, Di_8
- $Di_8 = < X, T \mid X^2 = I, T^8 = I, XTX = T^{-1} >$
- As circuit equalities,

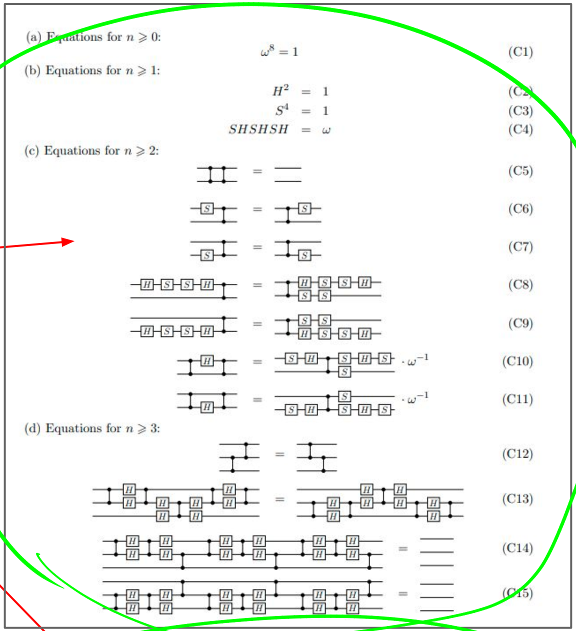
$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$



Circuit presentations

- Clifford circuits
- CNOT circuits — Lafont
- <CNOT, X, T> (CNOT-dihedral) circuits
- 2-qubit Clifford+T
- 3-qubit Toffoli+H⊗H = U(8,D) = Aut(E_8)
- 3-qubit Clifford+CS
- <H, CNOT, Rz(theta)> = U(2^n,C)
- Open questions:
  - ~~n-qubit Clifford+T?~~
  - n-qubit Toffoli+H?
  - n-qubit permutations



(a) Equations for $n \geqslant 0$:

$$\omega^8 = 1 \quad \text{(C1)}$$

(b) Equations for $n \geqslant 1$:

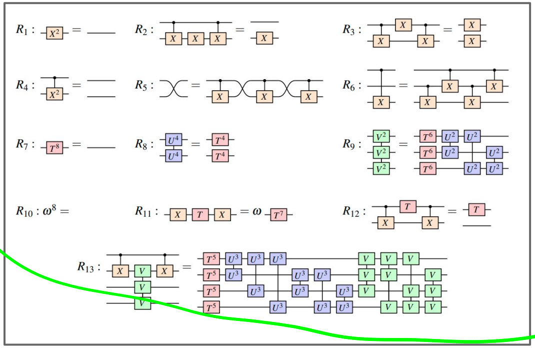
$$H^2 = 1 \quad \text{(C2)}$$

$$S^4 = 1 \quad \text{(C3)}$$

$$SHSHSH = \omega \quad \text{(C4)}$$

(c) Equations for $n \geqslant 2$:

(d) Equations for $n \geqslant 3$:



Equational theories & representations

- On their own not particularly useful
    - Complete equational theories typically involve going to exponential-size and -time normal forms
- Help us to understand the structure of gate sets
    - E.g. Circuits over <X, T> are isomorphic to the Dihedral group, hence have known properties
- Most useful when using a representation that elucidates (or mods out by) some relevant structure
    - E.g. Pauli exponentials, sum-over-paths, or the ZX-calculus

Example: Pauli exponentials

- Recall: Pauli group $P_n = \{i^{\{0,1,2,3\}}P_1 \otimes P_2 \otimes \ldots \otimes P_n\}$
- Recall: Clifford group $C_n = \{C \mid CP_nC^\dagger = P_n\}$
- Recall: Pauli exponential $R(\theta, P) = e^{i\theta P} = Ce^{i\theta(I\otimes I\otimes \ldots \otimes Z)}C^\dagger$

Proposition:

Any n-qubit Clifford+T circuit U with k T gates can be written as

$U=R(\pm\pi/4, P_1)R(\pm\pi/4, P_2)\ldots R(\pm\pi/4, P_k)C$ where C is Clifford

Computing the Pauli exponential rep

- Basic idea amounts to this: $CR(\theta,P) = R(\theta,CPC^{\dagger})C$ for C Clifford
- Explicitly,

$$e^{i\theta P} = \sum_{k=0}^{\infty} \frac{(i\theta)^k}{k!} P^k = \sum_{k=0}^{\infty} \frac{(i\theta)^{2k}}{2k!} I + \sum_{k=0}^{\infty} \frac{(i\theta)^{2k+1}}{2k+1!} P$$

$$= cos(\theta) I + i \sin(\theta) P$$

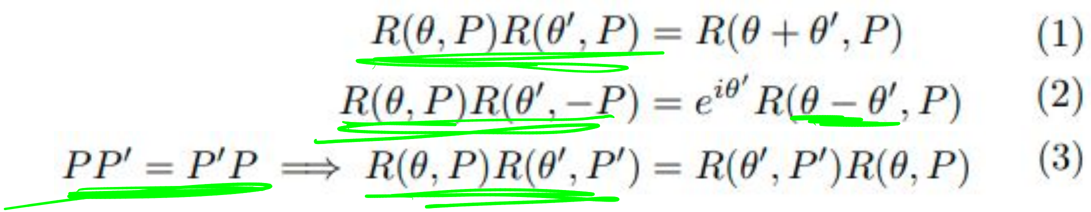
$$CPC^{\dagger}C$$
$$= (CPC^{\dagger})C$$

- Writing a Clifford+T circuit as $U = C_1T_1C_2T_2C_3\ldots C_kT_kC_{k+1}$ we can
  - Write every T gate as a Pauli exponential $R(\pm\pi/4, I^{i-1}\otimes Z\otimes I^{n-i-1})$, and
  - Commute all Clifford gates through to the right-hand side

Re-writing for T-count optimization

- Pauli exponentials effectively “mod out” by Cliffords

- Also have a simple (but incomplete) equational theory:



$$\underline{R(\theta, P)R(\theta', P)} = R(\theta + \theta', P) \qquad (1)$$

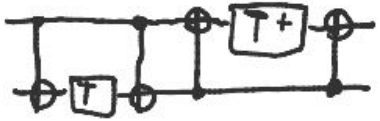
$$\underline{R(\theta, P)R(\theta', -P)} = e^{i\theta'} R(\underline{\theta - \theta'}, P) \qquad (2)$$

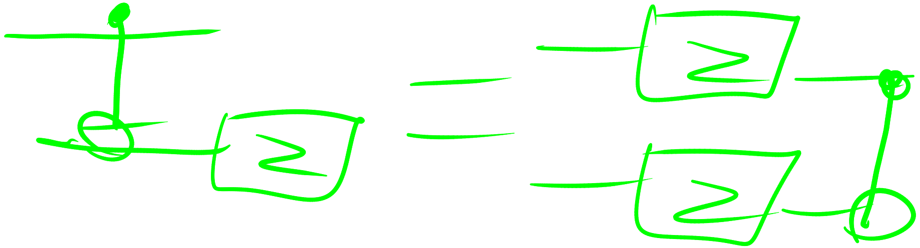
$$\underline{PP' = P'P} \implies \underline{R(\theta, P)R(\theta', P')} = R(\theta', P')R(\theta, P) \qquad (3)$$

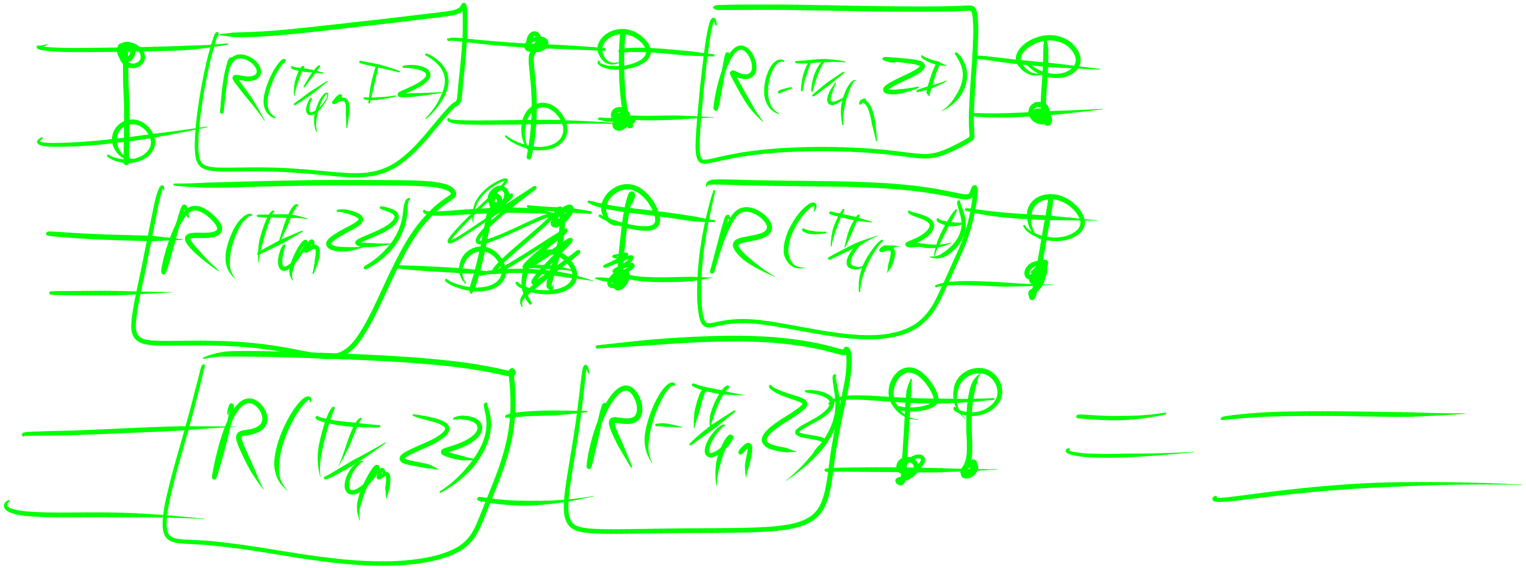
- Gives a simple T-count optimization procedure:
  - For each Pauli exponential:
    - Commute right with (3) until it can be merged with another by (1) or (2)
    - If it can’t be merged with any Pauli exponentials, leave where it is

Example

Recall:







Phase folding

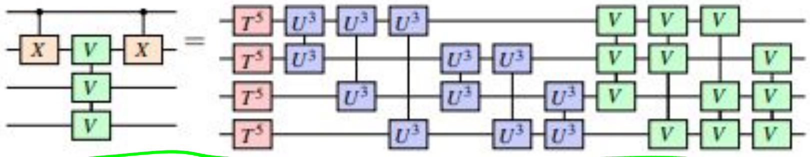
- The previous Pauli exponential optimization is an example of phase folding
- Basic idea: use commutation relations & gate cancellations to remove extraneous T gates (or other Z-axis rotations)
- Complexity? $O(|C|^2) \approx O(|\#T|^2)$

Theorem:

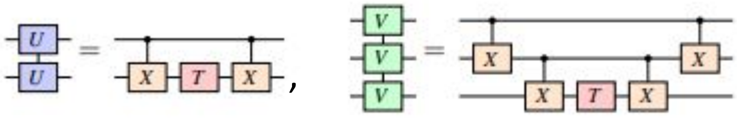
> *The Pauli-exponential optimization is optimal for the theory of Clifford equations, Clifford-T commutations, and $\underline{TT=S}$*

Is it optimal in general?

No! Incomplete theory (doesn’t capture Reed-Muller/spider nest identities)



where

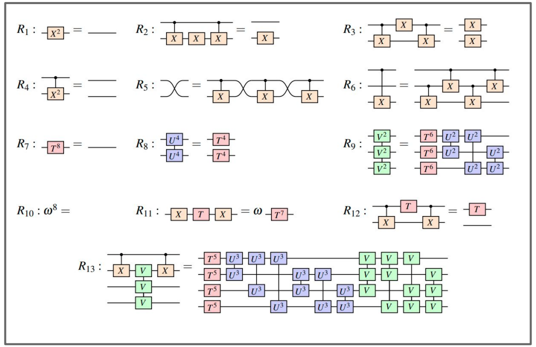


Sum of all 4-bit parities is 0 mod 8

In Pauli exponential form,

$$\prod_{P\in\{I,Z\}^{\otimes 4}} R\left(\frac{\pi}{4}, P\right) = I$$

Finding the optimal T-count



- Consider **just <CNOT, X, T> circuits**
- As pauli exponentials = strings of R(±π/4, P) where P is in $\{I,Z\}^{\otimes n}$
- All such Pauli exponentials commute, so it should be easy, right?
- Valid spider-nest equations in the Pauli exponential point of view = R13 “up to <CNOT, X>”, so valid equations are

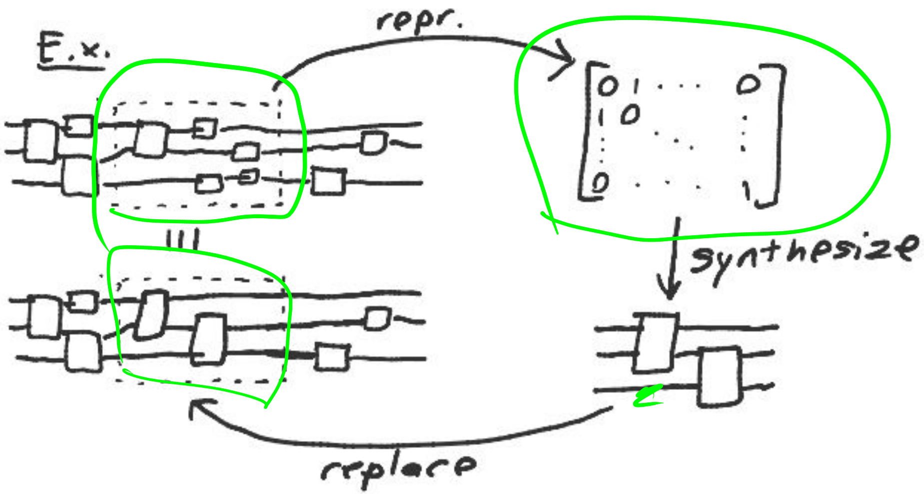
$$\prod_{P\in S\subseteq\{I,Z\}^{\otimes n}\text{s.t. }\dim\operatorname{span}(S)=4} R\left(\frac{\pi}{4},P\right)=I$$

Problem: no confluent, terminating, cost-decreasing re-write system!

Re-synthesis

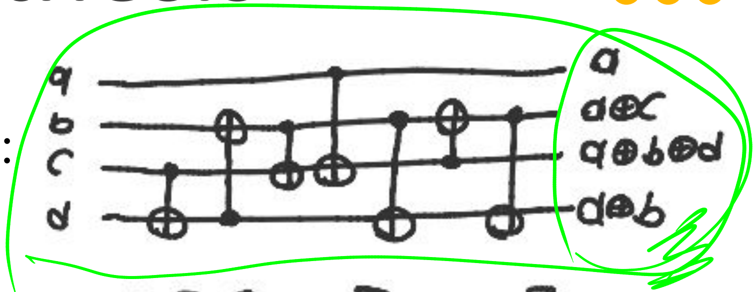
Re-synthesis based optimization

- Basic idea: compute some mathematical representation of a (sub-)circuit & synthesize optimally (or at least efficiently)



Example: <CNOT> resynthesis

- Re-synthesize (potentially in a larger circuit):

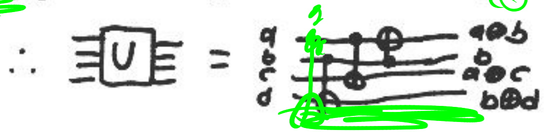


- First compute matrix representation:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} a \\ a\oplus c \\ a\oplus b\oplus d \\ a\oplus b \end{bmatrix}$$

- Next synthesize e.g. Patel-Markov-Hayes:

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \xrightarrow{R_1 \to R_1 + R_2} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \xrightarrow{R_3 \to R_1 + R_3} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \xrightarrow{R_4 \to R_2 + R_4} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Representations for re-synthesis

- Re-synthesis relies on understanding the mathematical essence of circuits
  - E.g. n-qubit <CNOT> = GL(n,F_2)
- Representation should (generally) be poly-time computable
- Synthesis should lead to good circuits by some metric
- A non-example:
  - A not-so great candidate is <CNOT + single qubit rotations> = U(n)
  - Matrix representation is exponential time to compute
  - Generic synthesis produces circuits of size O(4^n)
  - ⇒ Not (generally) a good candidate for re-synthesis!

CNOT-dihedral circuits

- Recall: CNOT-dihedral group (of order 8) = circuits over <CNOT, X, T>
- As a function of the computable basis, each gate only affects the state or phase (i.e. no basis change)

$$X : |x\rangle \mapsto |x \oplus 1\rangle \qquad CNOT : |x, y\rangle \mapsto |x, x \oplus y\rangle \qquad T : |x\rangle \mapsto e^{i\frac{\pi}{4}x}|x\rangle$$

Proposition:

Any circuit U over <CNOT, X, T> can be written as

$$U : |\vec{x} \in \mathbb{Z}_2^n\rangle \mapsto e^{i\frac{\pi}{4}\sum_{\vec{y}\in\mathbb{Z}_2^n} a_y \vec{x}\cdot\vec{y}}|A\vec{x}+\vec{b}\rangle$$

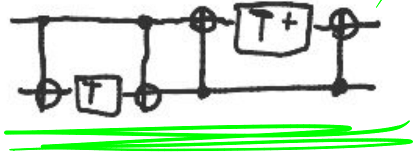
Dot product

“Phase polynomial”

Affine transformation

Example

$|x,y\rangle \longmapsto |x,y\rangle w \qquad w = e^{i\pi/4}$

- Our standard example, 

$$|x,y\rangle \xmapsto{CNOT_{1,2}} |x, x\oplus y\rangle \xmapsto{T_2} w^{x\oplus y}|x, x\oplus y\rangle$$

$$\xrightarrow{CNOT_{1,2}} w^{x\oplus y}|x,y\rangle \longrightarrow w^{x\oplus y}|x\oplus y, y\rangle \longrightarrow w^{\cancel{(x\oplus y)-(x\oplus y)}}|x\oplus y, y\rangle$$

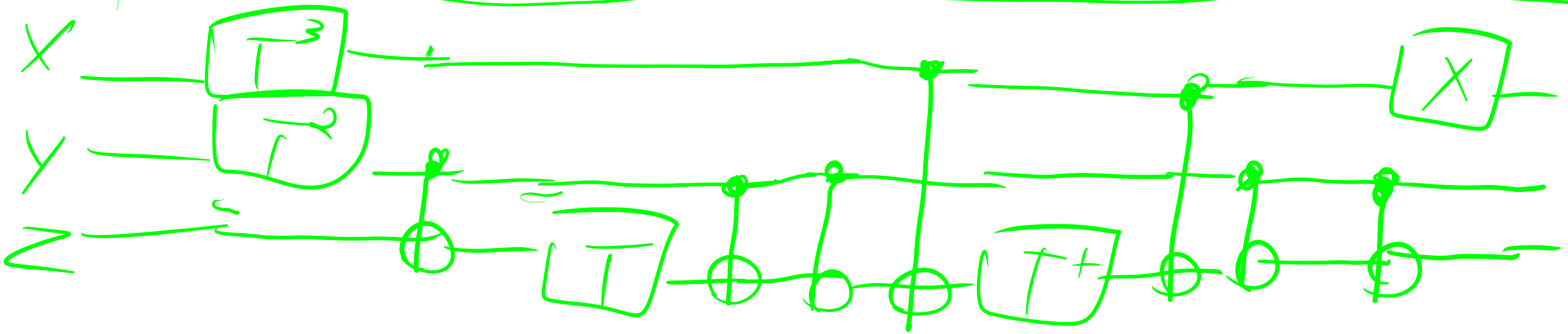
$$\longrightarrow |x,y\rangle$$

CNOT-dihedral re-synthesis

- We talked briefly about this already :)
- Each term $a_y\vec{x}\cdot\vec{y}$ of the phase polynomial is a rotation of T^{a_y} applied to a qubit in some parity $|\vec{x}\cdot\vec{y}\rangle = |x_{i_1}\oplus x_{i_2}\oplus\cdots\oplus x_{i_k}\rangle$ of the bits
- Ex. synthesize

$$|x, y, z\rangle \longmapsto |x\oplus 1, y, y\oplus z\rangle$$

$$U : |x, y, z\rangle \mapsto \omega^{3x+2y+(y\oplus z)-(x\oplus y\oplus z)}|x\oplus 1, y, y\oplus z\rangle \qquad \omega = e^{i\frac{\pi}{4}}$$



T-count optimal synthesis

- In the phase polynomial framework, spider nest identity is $\omega^{\sum_{\vec{y}\in\mathbb{Z}_2^4} \vec{x}\cdot\vec{y}} = 1$
  - ⇒ Exist distinct phase polynomials that give the same unitary!
- Idea: view an n-qubit phase polynomial as a length 2^n string of coefficients

$$\omega^{\sum_{\vec{y}\in\mathbb{Z}_2^n} a_y \vec{x}\cdot\vec{y}} \implies [a_0\ a_1\ \cdots\ a_{2^n}] \in \mathbb{Z}_8^{2^n}$$

- #T gates = # odd terms in this vector = hamming weight of binary residue
- Equivalent phase polynomials generate an equivalence relation on $\mathbb{Z}_8^{2^n}$

$$\vec{a}\sim\vec{b} \iff \vec{a}\in\vec{b}+C, \qquad C \triangleleft \mathbb{Z}_2^{2^n}$$

Reed-Muller characterization

Theorem:

The binary residue of C is equal to the (punctured) Reed-Muller code RM(n, n-4)

Implications:

- T-count optimization for <CNOT, T, X> equivalent to decoding RM(n, n-4)
- T-count upper bound of $O(n^2)$ $O(n^3)$
- $\mathbf{CNOTDih_{n,8}} = \mathbf{GA}(\mathbb{Z}_2, n) \ltimes \mathbb{Z}_8^{2^n} / \overline{\mathcal{RM}(n, n-4)}$

Phase polynomial synthesis problems

- The phase polynomial characterization of CNOT-dihedral circuits provides a lot of structure for studying synthesis problems

Cost metric	Complexity	Reduction	Lower bound	Upper bound
T-count	Believed NP-hard	Min-distance decoding of RM(n, 4-n)	Covering radius of RM(n, 4-n)	O(n^2)
T-depth	Poly-time	Matroid partitioning	O(1) w/ ancillas	O(1) w/ ancillas
CNOT-count	NP-hard in restricted cases	TSP/MLD	O(n^2)	O(n^2)

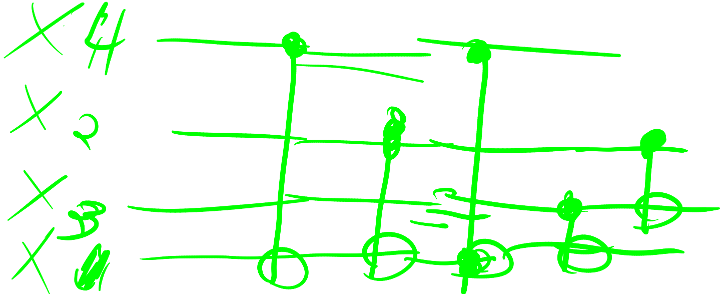
Poly-time heuristic “gray-synth” (Amy, Azimzadeh, Mosca “On the CNOT-complexity…”)

CNOT-minimizing synthesis

- Phase polynomial synthesis relies on computing each parity $\vec{x} \cdot \vec{y}, \quad a_y \neq 0$
- Computing parities is done with CNOT gates
- Synthesis problem:

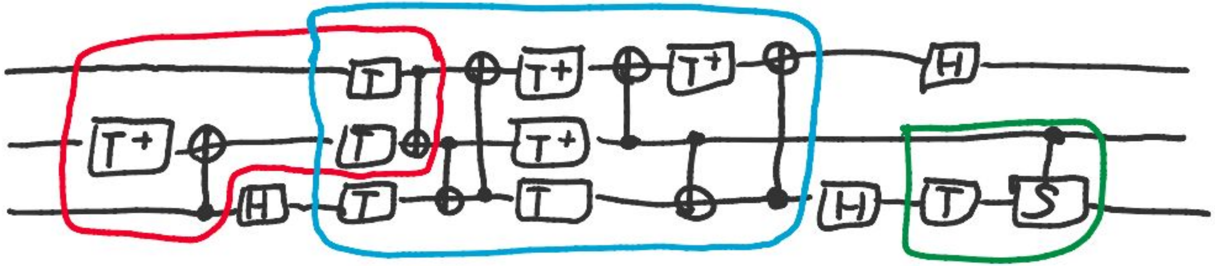
  Given a set S of parities of n bits, what is the minimal number of CNOTs needed to construct a tour of all parities in S?

- E.g. $\{\cancel{x_2 \oplus x_3}, \cancel{x_1}, \cancel{x_1 \oplus x_4}, \cancel{x_1 \oplus x_2 \oplus x_3}, \cancel{x_1 \oplus x_2 \oplus x_4}, \cancel{x_1 \oplus x_2}\}$



Choosing the right sub-circuit

- May be many ways of dividing up a circuit (so won’t get global optimum)



- What about other types of sub-circuits? E.g. Cliffords?

Representations of the Clifford group

- Recall: Clifford group <CNOT, S, H> permutes Paulis
  - ⇒ Clifford circuits can be represented as a permutation on P_n
- Optimization: action is a linear permutation, so similar to CNOT circuits, can represent efficiently by its action on 2n generators of the Pauli group
- Problem: synthesis problem doesn't map directly to Gaussian elimination
- Solution: use the sum-over-paths/affine representation

From phase polynomials to SOP

- Can extend the phase polynomial representation to Clifford+T circuits using

$$H : |x\rangle \mapsto \frac{1}{\sqrt{2}} \sum_{y\in\mathbb{Z}_2} (-1)^{xy}|y\rangle$$

"Path variable"

Proposition:

Any circuit U over Clifford+T gates can be represented as a sum-over-paths

$$U : |\vec{x} \in \mathbb{Z}_2^n\rangle \mapsto \frac{1}{\sqrt{2^k}} \sum_{\vec{y}\in\mathbb{Z}_2^k} \omega^{P(\vec{x},\vec{y})}|A(\vec{x},\vec{y})+\vec{b}\rangle$$

"Real" polynomial

Re-writing + phase polynomials

- The sum-over-paths is not unique:

$$I = HH : |x\rangle \mapsto \frac{1}{2} \sum_{y,z \in \mathbb{Z}_2} (-1)^{xy+yz} |z\rangle$$

- But we can simplify by re-writing the sum-over-paths

$$\sum_{\vec{x},y} e^{iQ(\vec{x})} \, |f(\vec{x})\rangle \longrightarrow_{\text{Cliff}} 2 \sum_{\vec{x}} e^{iQ(\vec{x})} \, |f(\vec{x})\rangle \tag{E}$$

$$\sum_{\vec{x},y,z} (-1)^{y(z \oplus P(\vec{x}))} e^{iQ(\vec{x},z)} \, |f(\vec{x},z)\rangle \longrightarrow_{\text{Cliff}} 2 \sum_{\vec{x}} e^{iQ(\vec{x},\overline{P}(\vec{x})} \, |f(\vec{x},P(\vec{x}))\rangle \tag{H}$$

$$\sum_{\vec{x},y} i^{y} (-1)^{yP(\vec{x})} e^{iQ(\vec{x})} \, |f(\vec{x})\rangle \longrightarrow_{\text{Cliff}} \omega\sqrt{2} \sum_{\vec{x}} (-i)^{\overline{P}(\vec{x})} e^{iQ(\vec{x})} \, |f(\vec{x})\rangle \tag{ω}$$

Example

- Recall: SHSHSH = ωI

Clifford normalization

- A Clifford sum-over-paths has the form

$$|\vec{x}\rangle = \frac{\omega^l}{\sqrt{2^k}} \sum_{\vec{y}\in\mathbb{Z}_2^k} i^{L(\vec{x},\vec{y})}(-1)^{Q(\vec{x},\vec{y})}\,|f(\vec{x},\vec{y})\rangle$$

linear

quadratic

affine

Proposition (affine representation):

The re-write system* $\rightarrow_{\text{Cliff}}$ terminates on a Clifford sum-over-paths in polynomial time with a unique normal form called the affine representation

$$|\vec{x}\rangle = \frac{\omega^l}{\sqrt{2^k}} \sum_{\vec{y}\in\mathbb{Z}_2^k} i^{L(\vec{x},\vec{y})}(-1)^{Q(\vec{x},\vec{y})}\,|\vec{y}\rangle \otimes |f(\vec{x},\vec{y})\rangle$$

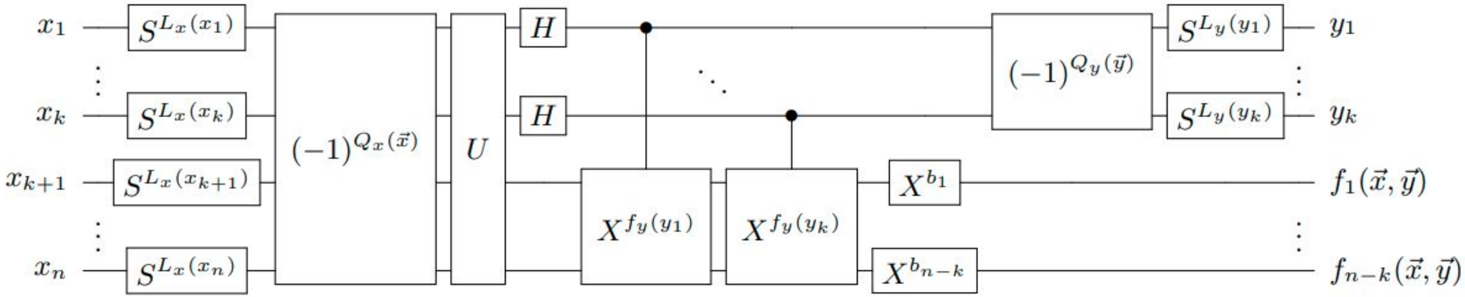
Clifford re-synthesis

- Compute (in poly-time) the sum-over-paths for a Clifford (sub-)circuit
- Normalize to the affine representation

$$|\vec{x}\rangle = \frac{\omega^l}{\sqrt{2^k}} \sum_{\vec{y}\in\mathbb{Z}_2^k} i^{L(\vec{x},\vec{y})}(-1)^{Q(\vec{x},\vec{y})} |\vec{y}\rangle \otimes |f(\vec{x},\vec{y})\rangle$$

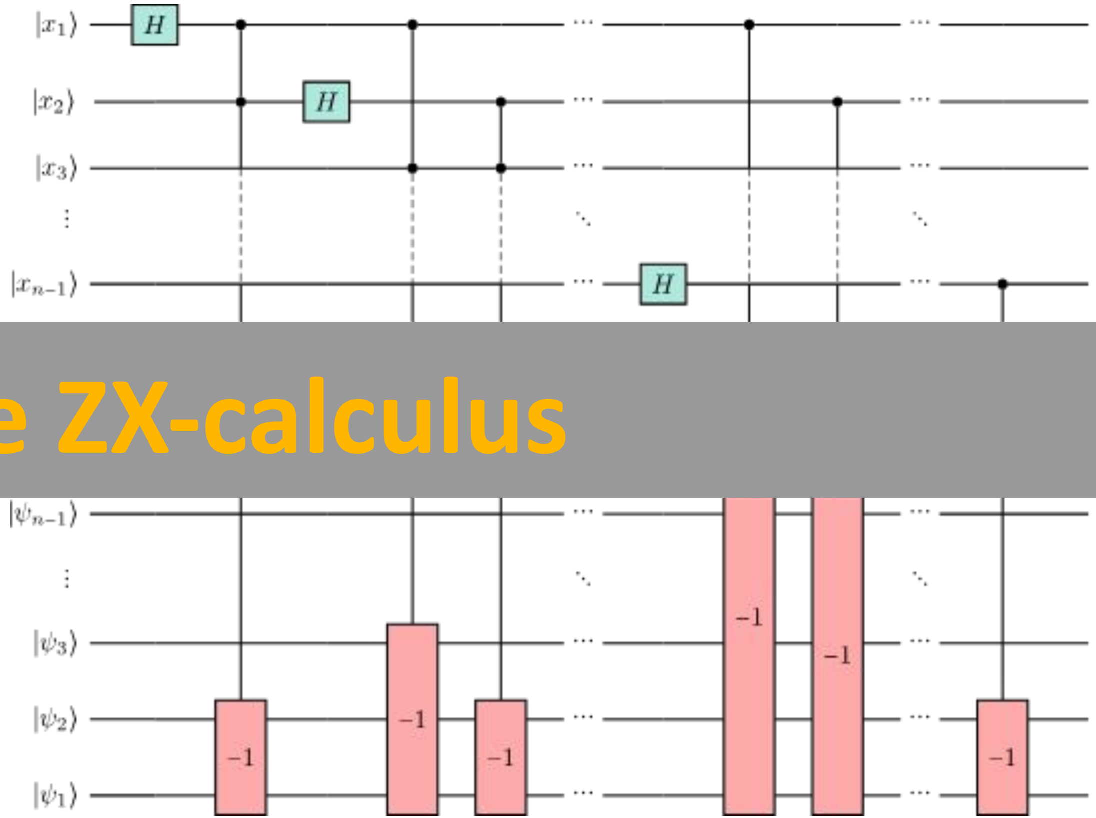
- The affine representation factors into $\omega^l PV(\mathrm{H}^{\otimes k} \otimes \mathrm{I}_{n-k})UD$ where
  - D implements the phase terms conditional on the input |x>
  - P implements the phase terms condition on the output |y>
  - H produces the sum-over-paths indexed by y
  - V sends |x>|f(x, 0)> to |x>|f(x,y)>
  - U is a binary linear permutation defined by $U = \omega^{-l}(\mathrm{H}^{\otimes k} \otimes \mathrm{I}_{n-k})V^{\dagger}P^{\dagger}\Psi D^{\dagger}$

Properties of the Clifford normal form



- Minimal H-count & H-depth
  - ⇒ Important as the CNOT-dihedral T-count bound implies $O(hn^2)$ T gate upper bound over Clifford+T where h is the H-depth
- Reduces synthesis of Clifford circuits to synthesis of U (i.e. CNOT circuits)

The ZX-calculus



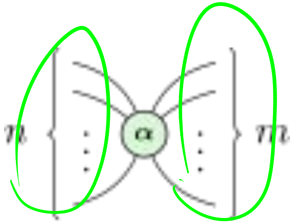
The ZX-calculus

- So far we’ve talked about two representations useful for optimization:
  - Pauli exponentials
  - Phase polynomials/sum-over-paths
- Their effectiveness lies in non-uniqueness coupled with rewrite rules
  - Uniqueness for Clifford+T implies not poly-time computable
  - Rewrite rules imply the possibility of optimization
- A complementary representation with similar properties is the **ZX-calculus**
  - In fact, all methods discussed today have equivalent formulations in the ZX-calculus

ZX diagrams

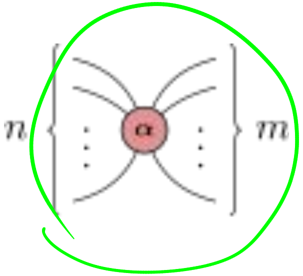
- “Generalized circuits” or tensor networks
- ZX-diagram is a graph with two types of nodes: Z and X **spiders**

- Spiders with n outgoing edges correspond to 2^n-dimensional tensors



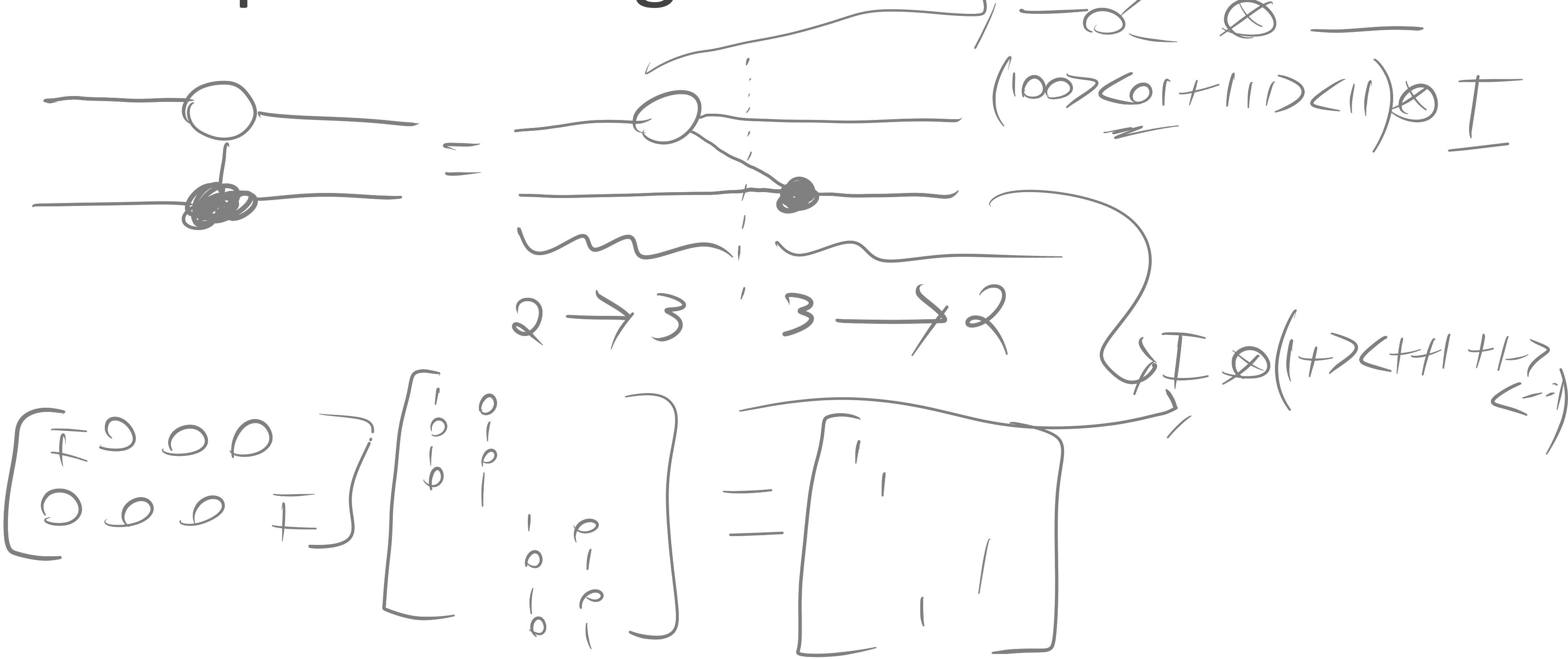
$$= \underline{|0\rangle^{\otimes m}\langle 0|^{\otimes n}} + \underline{e^{i\alpha}|1\rangle^{\otimes m}\langle 1|^{\otimes n}}$$

$2^n \rightarrow 2^n$



$$= |+\rangle^{\otimes m}\langle +|^{\otimes n} + e^{i\alpha}|-\rangle^{\otimes m}\langle -|^{\otimes n}$$

Example: CNOT gate



$|+\rangle\langle+| = \frac{1}{2\sqrt{2}} |0\rangle\langle 00| + |0\rangle\langle 01|$

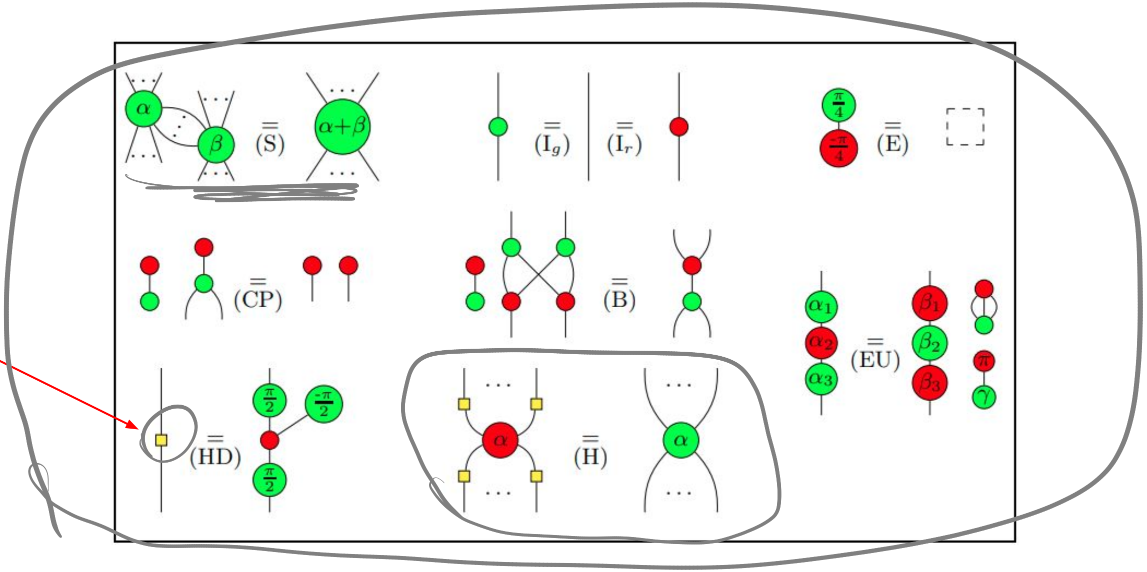
$(|00\rangle\langle 0| + |11\rangle\langle 1|) \otimes I$

$I \otimes (|+\rangle\langle+| + |-\rangle\langle-|)$

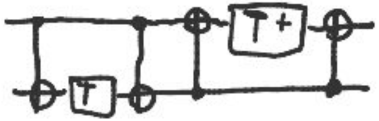
Re-writing ZX diagrams

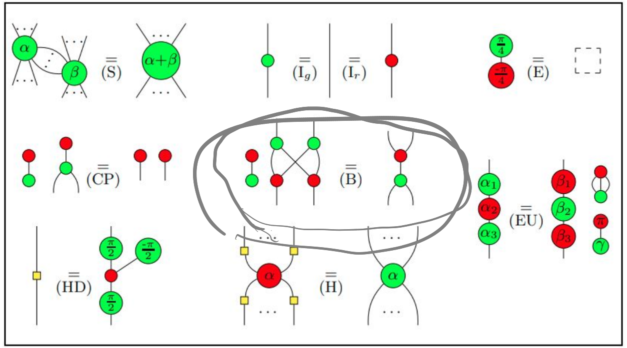
- Basic principle: “only connectivity matters”
  - I.e. it doesn’t matter how you draw the graph, it gives you the same tensor up to isomorphism
- The ZX-calculus comprises a (complete) equational theory on ZX diagrams

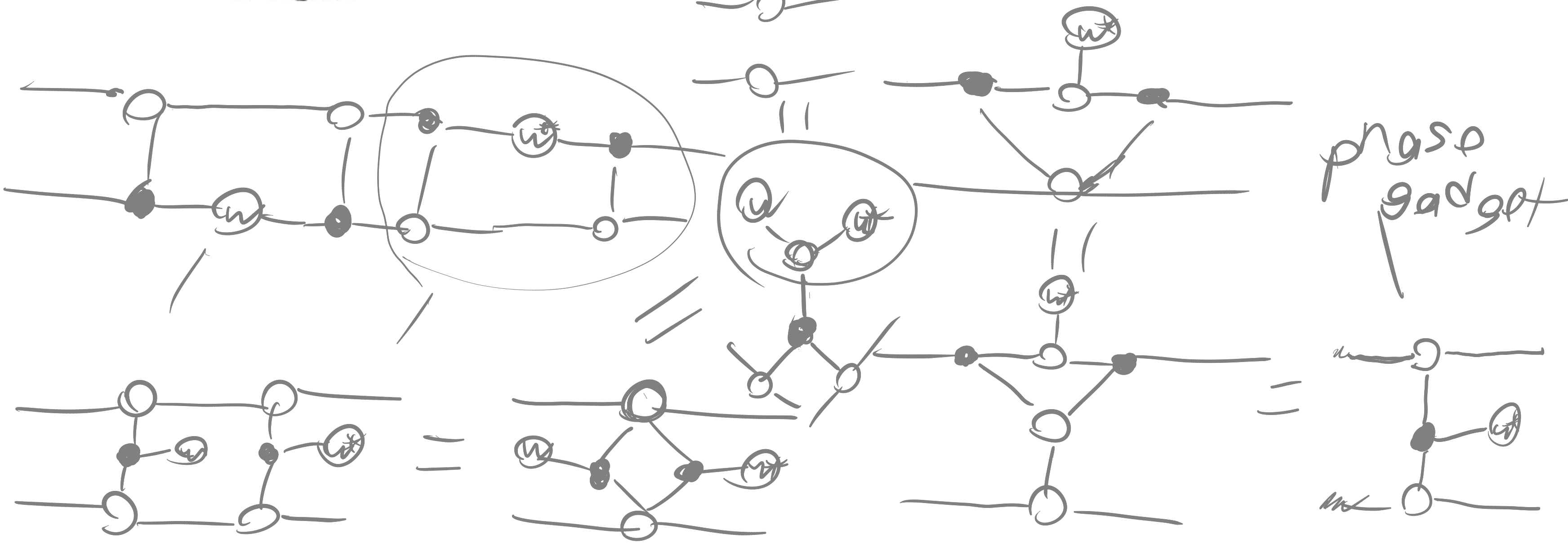
Hadamard



Example

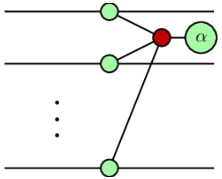
- Let's do  one last time





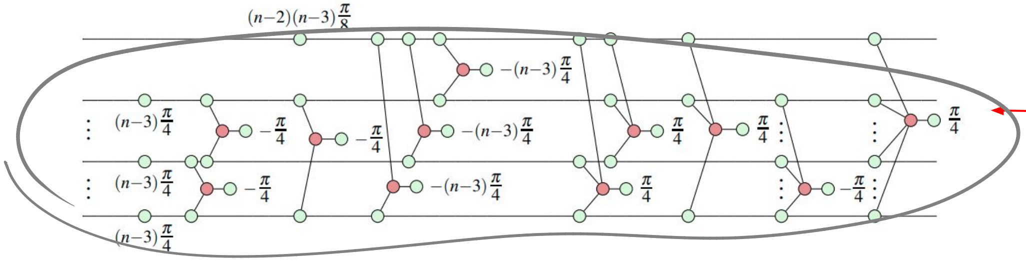
Phase gadgets

- Diagrams of the form



are called phase gadgets

- Phase gadgets are exactly terms of a phase polynomial
  - phases conditional on a parity of some selection of bits)
- Using phase gadgets, can do all the same optimizations as with phase polynomials/pauli exponentials



Spider nest equation

So what's the upshot?

- ZX-calculus is a complete equational theory
  - Unlike Pauli exponentials/sum-over-paths
- Completeness makes it useful as a theorem-proving tool
  - There always exists a manual proof of equivalence/optimization
- Drawback to this power is difficulty automating reasoning
  - Rules are not obviously directed
  - In comparison, the sum-over-paths has directed (but incomplete) rules
  - Still, can find effective normalization procedures in ZX for, e.g., Clifford circuits

Comparing representations

Pauli exponentials	**Sum-over-paths**	**ZX-calculus**
Pauli exponential	Term of phase polynomial	Phase gadget
Commuting string of Pauli exponentials	Phase polynomial	Adjacent phase gadgets
Equivalent strings of commuting Paulis	Reed-Muller identities	Spider nest equations
Commuting Cliffords to the end	$\rightarrow_{\text{Cliff}}$	Clifford normalization (pivoting + complementation)
Incomplete equational theory	Incomplete **(but strictly larger)** equational theory	Complete equational theory

Readings for next week

- Posted to the website
  - Xu et al., *Quartz: Superoptimization of Quantum Circuits*. arXiv:2204.09033
  - Duncan, Kissinger, Perdrix, van de Wetering, *Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus*. arXiv:1902.03178
  - Häner, Hoefler, Troyer, *Assertion-Based Optimization of Quantum Programs*. arXiv:1810.00375
  - ~~Heyfron, Campbell, *An Efficient Quantum Compiler that reduces T count*. arXiv:1712.01557~~
  - ~~Amy, Maslov, Mosca, *Polynomial-time T-depth Optimization of Clifford+T circuits via Matroid Partitioning*. arXiv:1303.2042~~
    - We don’t have time to discuss these two, but references for phase polynomial techniques

- As before send me a short (paragraph or two) summary of **ONE (1)** paper of your choice before next class and be prepared to give a short summary of any of the papers in class