Circuit optimization

Up until now we've mostly considered the problem of synthesizing or compiling good circuits. What if we already have a bad circuit though? Improving upon this problem is the circuit optimization problem.

Problem (circuit optimization)

Given a circuit C over a gate set G and cost function $c : Circuits(G) \to \mathbb{R}$, find some C' over G s.t.

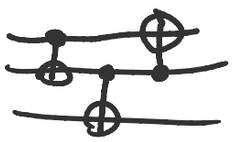
1. $C' \equiv C$ (i.e. they implement the same operator)
2. $c(C') < c(C)$ (ideally C' minimizes c...)

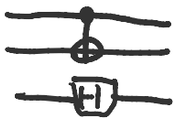
(cost functions)

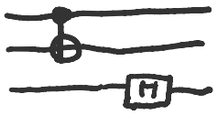
- Quantum cost (# of gates over $\{X, CX, CCX\}$)
  this is old and irrelevant now. don't use please
- T-count (# T gates)
  primary metric for FT computations over Clifford+T
- CNOT-count (# CNOT/CX gates)
  Often used for physical computations since entanglement is expensive & error prone.
- depth (roughly, the "length" of the circuit)
  governs (roughly) the time to execute, and by extension the error in non-FT computations
- T-depth (the number of state injection stages)
  This is an older metric based on a number of assumptions, notably that (1) Clifford circuits are performed in one code cycle, and (2) space is effectively unbounded. Mostly replaced with more nuanced cost models now

Circuit depth

Ex.

What is the depth of  ? 3

what about  ? 1

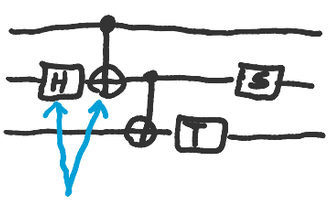
or  ? ?

Circuit depth for circuits which have not been routed or mapped to a specific hardware & execution plan is typically taken as the minimum depth in this regard.

If we view the circuit as a directed acyclic graph with edges from each gate output to the input of the next gate (i.e. a dependence graph of the circuit), then the minimum depth is the length of a longest path, called a critical path.

E.g.

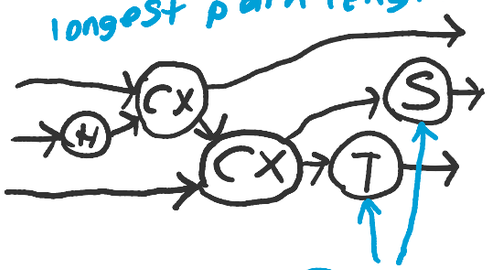
Circuit



CNOT here depends on the output of H

dependency graph

longest path length = 4



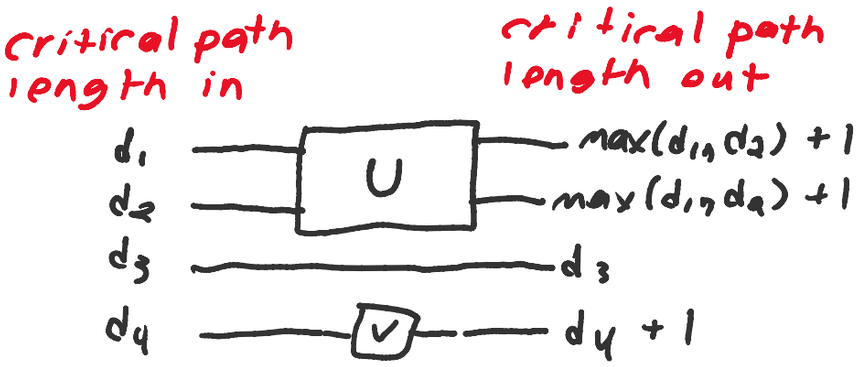
S has no dependence on T

T-depth is likewise the maximum T-count along any path in the graph.

Depth computation

Depth (or T-depth) can be calculated by updating the critical path length for each qubit after each gate, then taking the overall maximum

i.e.



Or in pseudo-code,

```
1. depths[i] = 0 ∀ i
2. for gate in circuit
3.     d = max_{i ∈ inputs(gate)} depths[i]
4.     depths[i] = d + 1  ∀ i ∈ outputs(gate)
5. return max_i depths[i]
```

Circuit optimization algorithms

Compared to classical compilor optimizations, quantum circuit optimizations are (currently) more limited in form. Part of this is due to the relative simplicity of the circuit model with respect to control flow...

(Broad catogories of opt. algorithms)

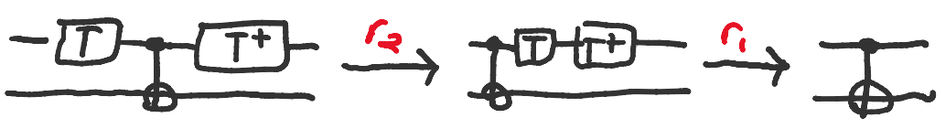
quantum	classical analogue
template/rewrite based	peephole optimization
re-synthesis	?
analysis-based	dataflow optimization (e.g. constant propagation) abstract interpretation basically everything

*most opt. algorithms fit into these...
main analysis-based opt. is phase-folding

(Template/re-write based)
- parliest algorithms from Dmitri Maslov & co, circa 2000's

Basic idea:
Given a gate set G and a set of circuit equalities over G, repeatedly perform re-writes of a circuit C using those equalities.

E.g. Given 

Then 

Re-write based optimization

(Generic method)

Given a circuit C & set of rewrite rules $C_1 \to C_2$,

1. For each gate g_i in $C = g_1 \cdots g_k$
   2. See if $g_i \cdots g_j$ matches some rule $C_1 \to C_2$ and if so, replace $g_i \cdots g_j$ with C_2
3. Repeat until no more changes can be made

Complexity: $O(k^2 \cdot m \cdot n)$ where m is # rules, n is max length of lhs

(assuming no cycles...)

(Designing re-write rules)

Designing effective re-write based optimizations generally boils down to finding a good set of re-write rules. Factors include:

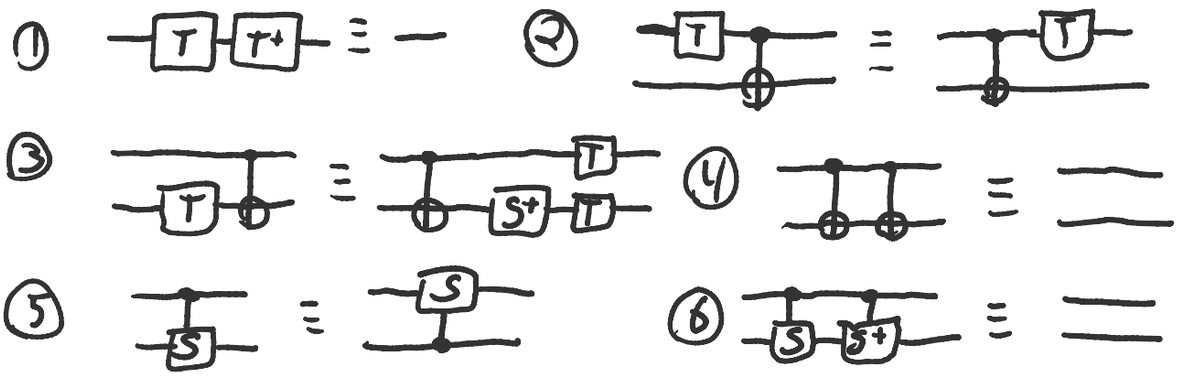
- Size - smaller = more efficient
- decreasing - if re-write rules are not decreasing with respect to the cost function, i.e. $\forall C_1 \to C_2$, $c(C_1) \geq c(C_2)$, then cost can increase. On the other hand, more likely to get stuck in local minima w/ a small decreasing set of rules
- completeness - is every equivalent circuit reachable via the given re-write rules?
- confluence - does the order matter?

In practice, re-writing is generally not very effective and is usually implemented as basic simplifications (e.g. $UU^\dagger = I$) and commutations (e.g. $(Z \otimes I)CX = CX(Z \otimes I)$)

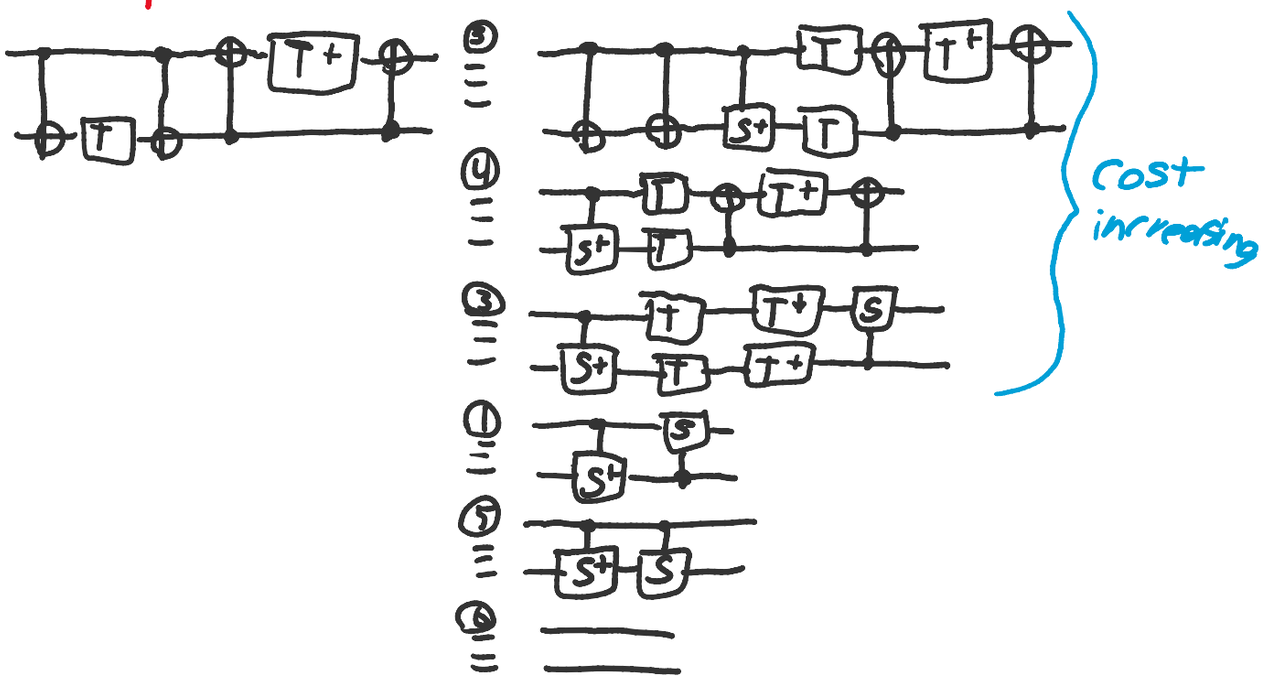
But the completeness question is interesting and relevant to upcoming material ☺

Example

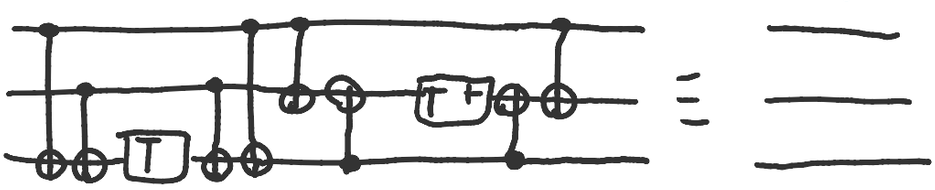
Consider the following set of equalities (bi-directional re-write rules)



We can optimize the following circuit as follows:



To avoid the use of cost-increasing re-writes, could add  as an equality. Doing so leads to over specialized rules. For instance, the following equality holds for the same reason, but can't be easily found using (7).



can be found with (7) and some CNOT rules...

Equational (circuit) theories

More formally, we can think of circuits (over G) as **words** or **terms** of a (free) algebra $(G, \cdot, \otimes)$. That is, a circuit C is:

- A gate $g \in G$
- A (well-formed) sequential composition $C_2 \cdot C_1$, or
- A parallel composition $C_1 \otimes C_2$

The **standard model** or **semantics** of a circuit C, denoted $[\![C]\!]$, is a linear operator on $\mathbb{C}^{2^n}$. We can define $[\![\cdot]\!]$ inductively as

- $[\![g]\!] = U_g \quad \forall g \in G$
- $[\![C_2 \cdot C_1]\!] = [\![C_2]\!][\![C_1]\!]$
- $[\![C_1 \otimes C_2]\!] = [\![C_1]\!] \otimes [\![C_2]\!]$

Ex.

technically should have arities...

Let $G = \{g_1, g_2, g_3\}$.

We define the semantics of $(G, \cdot, \otimes)$ by giving a linear operator for each $g \in G$.

Say $[\![g_1]\!] = \begin{bmatrix} 0 & i \\ i & 0 \end{bmatrix}$, $[\![g_2]\!] = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$, $[\![g_3]\!] = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

Then

$$[\![g_2 g_3]\!] = [\![g_2]\!][\![g_3]\!] = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ i & 0 \end{bmatrix}$$

Equational theories

Given a gate set G, a set $R \subseteq \langle G \rangle \times \langle G \rangle$ of **relations** defines an **equational theory** where we say that R **proves** that $C_1 \equiv C_2$, written $th_R \vdash C_1 \equiv C_2$, if a proof can be built from the following inference rules:

$$\frac{(C_1, C_2) \in R}{th_R \vdash C_1 \equiv C_2} \text{ (R-rules)}$$

$$\frac{th_R \vdash C_2 \equiv C_1}{th_R \vdash C_1 \equiv C_2} \text{ (comm)} \qquad \frac{th_R \vdash C_1 \equiv C_2 \quad th_R \vdash C_2 \equiv C_3}{th_R \vdash C_1 \equiv C_3} \text{ (trans)}$$

$$\frac{}{th_R \vdash C \equiv C} \text{ (refl)} \qquad \frac{th_R \vdash C_2 \equiv C_2' \quad \Delta \in \{\cdot, \otimes\}}{th_R \vdash C_1 \Delta C_2 \Delta C_3 \equiv C_1 \Delta C_2' \Delta C_3} \text{ (sub terms)}$$

(informally, if you can get C_2 from C_1 by applying rewrites from R in the obvious way)

(Soundness)

A theory th is **sound** if whenever $th \vdash C_1 \equiv C_2$, then $[\![C_1]\!] = [\![C_2]\!]$

(informally, every equality $C_1 \equiv C_2 \in R$ is true)

(Completeness)

A theory th is **complete** if whenever $[\![C_1]\!] = [\![C_2]\!]$, then $th \vdash C_1 \equiv C_2$

(informally, every true equality $C_1 \equiv C_2$ can be proven)

(Aside)

In the circuit-world, we often talk about the equivalent **algebraic** notion of a **presentation** by generators & relations.

Complete theories

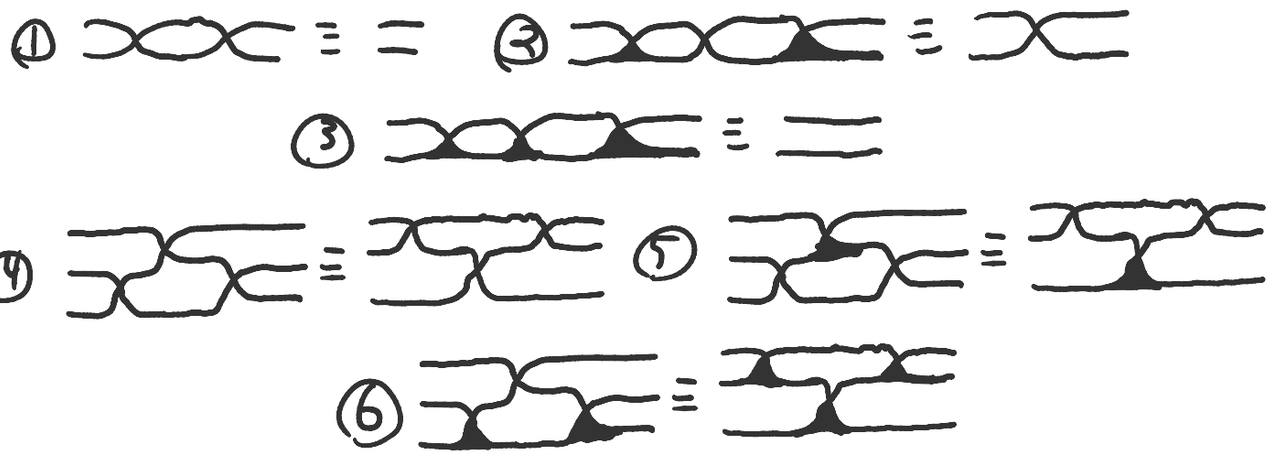
Known complete equational theories are

- Clifford circuits
- CNOT-dihedral circuits
  (circuits over $\{H, X, CNOT, R_z(\pi/2^k)\}$)
- CNOT-circuits (below)
- 1-2-qubit Clifford+T circuits
- Circuits over $\{H, CNOT, SWAP, e^{i\theta}, R_z(\theta)\}$
  ← very recent result →

Ex. (Lafont 2003)

(This is a fascinating, non-quantum paper)
"Towards an algebraic theory of Boolean circuits"

The following relations are complete for $\{SWAP, CNOT \cdot SWAP\}$
($CNOT \cdot SWAP = \begin{smallmatrix} x \\ y \end{smallmatrix} \times \begin{smallmatrix} y \\ x \oplus y \end{smallmatrix}$)



Open questions

- Relations for Toffoli + Hadamard
- Relations for n-qubit Clifford+T

Normal forms

A set of relations R gives rise to a re-writing system defined by dropping (refl) & (comm) and denoted $th_R \vdash C_1 \to C_2$

(Completeness)

A re-writing theory is complete if whenever $[\![C_1]\!] = [\![C_2]\!]$, $\exists C_3$ s.t. $th_R \vdash C_1 \to C_3$, $th_R \vdash C_2 \to C_3$

(Normal form)

A circuit C is in normal form if $\nexists C'$ such that $th_R \vdash C \to C'$

(Uniqueness)

Normal forms are said to be unique if for any normal forms $C_1 \neq C_2$, $[\![C_1]\!] \neq [\![C_2]\!]$

(Note: this is stronger than the notion in abstract re-writing)

Fact

If a re-writing theory/system has unique normal forms for every circuit, it is complete. Moreover, if every $C_1 \to C_2 \in R$ is cost non-increasing, these normal forms are cost optimal

Thm (Matsumoto-Amano)

There exists a re-write system which is complete and T-count optimal for single-qubit Clifford+T.