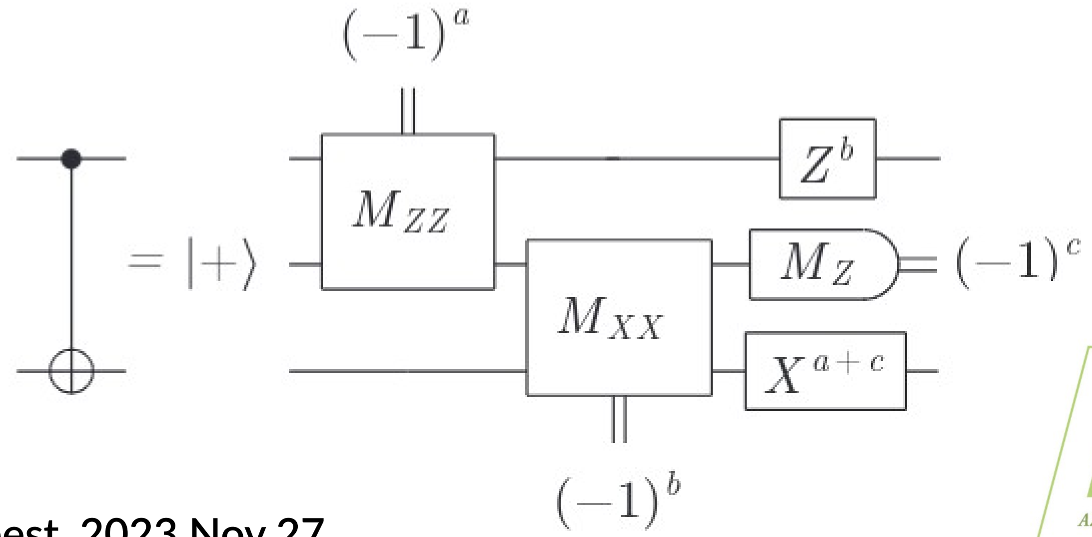
How to simulate Clifford circuits

Danniel Gottesman, 1998:
The Heisenberg Representation of Quantum Computers

János K. Asbóth[1,2]

1: Budapest University of Technology and Economics

2: Wigner Research Centre for Physics, Budapest



QuSZIT Seminar, Budapest, 2023 Nov 27

Nemzeti Kutatási, Fejlesztési és Innovációs Hivatal

AZ INNOVÁCIÓ LENDÜLETE

AZ NKFI ALAPBÓL MEGVALÓSULÓ PROJEKT

In Quantum Mechanics, we have Schrödinger and Heisenberg Pictures (and various interaction pictures)

$$\hat{U}(t) = \mathbb{T}e^{-i\int_0^t \hat{H}(t')dt'} \approx e^{-i\hat{H}(t_N)dt}e^{-i\hat{H}(t_{N-1})dt}\dots e^{-i\hat{H}(t_2)dt}e^{-i\hat{H}(t_1)dt}$$

Schrödinger

$$i\hbar\frac{\partial}{\partial t}|\psi(t)\rangle = \hat{H}|\psi(t)\rangle$$

$$|\psi(t)\rangle = U(t)|\psi(0)\rangle$$

$$\langle\hat{A}\rangle = \langle\psi(t)|\,\hat{A}\,|\psi(t)\rangle$$

$$\langle\hat{A}\rangle = \langle\psi(0)|\,\hat{U}^\dagger\hat{A}\hat{U}\,|\psi(0)\rangle$$

Heisenberg

$$\frac{d}{dt}A_{\mathrm{H}}(t) = \frac{i}{\hbar}[H_{\mathrm{H}}, A_{\mathrm{H}}(t)] + \left(\frac{\partial A_{\mathrm{S}}}{\partial t}\right)_{\mathrm{H}}$$

$$\hat{A}_H(t) = \hat{U}^\dagger(t)\hat{A}(0)\hat{U}(t)$$

solution of $\hat{U}\hat{A}_H = \hat{A}_S\hat{U}$

Acting with A after time evolution is equivalent to acting with what before time evolution?

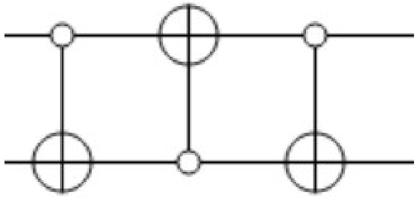
The Heisenberg picture can also be applied to quantum computers (quantum circuits)

$$\hat{U}(t) = \mathbb{T}e^{-i\int_0^t \hat{H}(t')dt'} \approx e^{-i\hat{H}(t_N)dt}e^{-i\hat{H}(t_{N-1})dt}\dots e^{-i\hat{H}(t_2)dt}e^{-i\hat{H}(t_1)dt}$$

$$\hat{A}(t) = \hat{U}^\dagger(t)\hat{A}(0)\hat{U}(t)$$

What does this circuit do? ~~To input states~~ To a complete set of operators?

$$\hat{U}(t) = e^{-i\hat{H}_3 dt}e^{-i\hat{H}_2 dt}e^{-i\hat{H}_1 dt} =$$



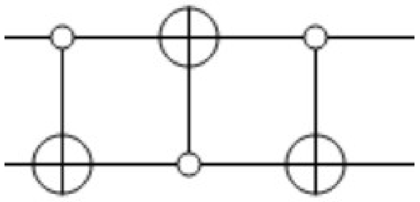
Quantum computer scientists prefer to think about the Heisenberg picture a little differently

$$\hat{U}(t) = \mathbb{T}e^{-i\int_0^t \hat{H}(t')dt'} \approx e^{-i\hat{H}(t_N)dt}e^{-i\hat{H}(t_{N-1})dt}\dots e^{-i\hat{H}(t_2)dt}e^{-i\hat{H}(t_1)dt}$$

$$\hat{A}_G = \hat{U}(t)\hat{A}\hat{U}^\dagger(t)$$

Acting with A before U corresponds to acting with A_G after U.

$$\hat{U}(t) = e^{-i\hat{H}_3 dt}e^{-i\hat{H}_2 dt}e^{-i\hat{H}_1 dt} =$$



“Complete set of operators” = X and Z on every qubit. This generates all Pauli strings

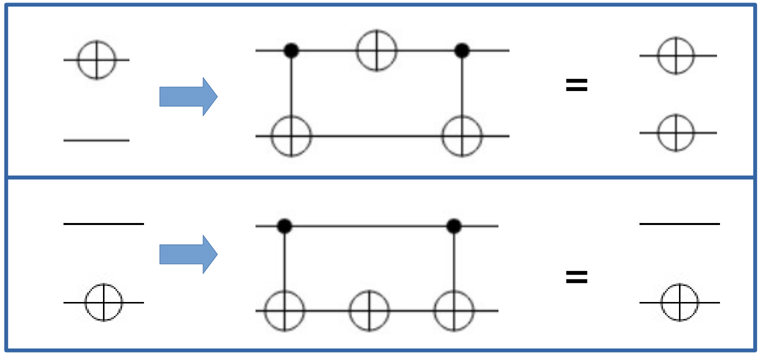
$$\mathcal{P} = \{\text{all Pauli strings, e.g.} \quad \hat{X}_1\hat{Y}_4\hat{Z}_5\hat{Z}_6\}$$

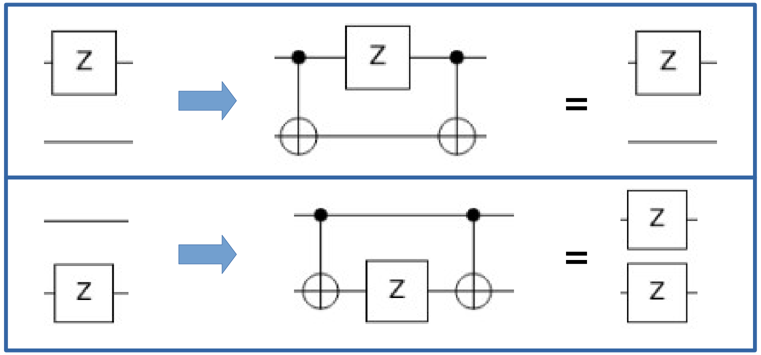
Clifford operators (gates): those that transform Pauli strings to Pauli strings (not superposition of Pauli strings)

As an example, CNOT is the mapping:

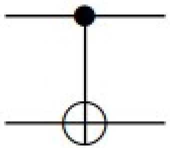
What does a CNOT do? ~~To input states~~ To a complete set of operators?

$$\hat{A} \Rightarrow \hat{U}(t)\hat{A}\hat{U}^\dagger(t)$$





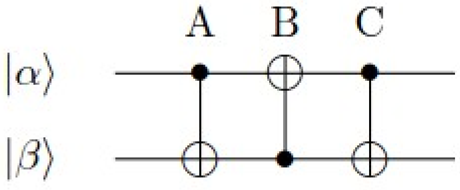
$$\begin{aligned}
X \otimes I &\rightarrow X \otimes X \\
I \otimes X &\rightarrow I \otimes X \\
Z \otimes I &\rightarrow Z \otimes I \\
I \otimes Z &\rightarrow Z \otimes Z
\end{aligned}$$



We track Pauli X and Z operators to find out that the curcuit posed above … is a SWAP

$$
\begin{aligned}
X \otimes I &\rightarrow X \otimes X \\
I \otimes X &\rightarrow I \otimes X \\
Z \otimes I &\rightarrow Z \otimes I \\
I \otimes Z &\rightarrow Z \otimes Z
\end{aligned}
$$

a)



b)

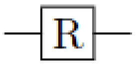
A: CNOT$(1 \rightarrow 2)$	$\overline{X}_1$	$X \otimes X$
	$\overline{X}_2$	$I \otimes X$
	$\overline{Z}_1$	$Z \otimes I$
	$\overline{Z}_2$	$Z \otimes Z$
B: CNOT$(2 \rightarrow 1)$	$\overline{X}_1$	$I \otimes X$
	$\overline{X}_2$	$X \otimes X$
	$\overline{Z}_1$	$Z \otimes Z$
	$\overline{Z}_2$	$Z \otimes I$
C: CNOT$(1 \rightarrow 2)$	$\overline{X}_1$	$I \otimes X$
	$\overline{X}_2$	$X \otimes I$
	$\overline{Z}_1$	$I \otimes Z$
	$\overline{Z}_2$	$Z \otimes I$

Figure 1: Alice's quantum computer: a) network, b) analysis.

All the basic relations needed to track gates

R

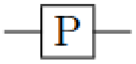
$$X \to Z$$
$$Z \to X$$



new name $\hat{H} = \hat{X} + \hat{Z} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$

P

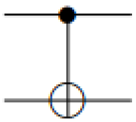
$$X \to Y$$
$$Z \to Z$$



new name $\hat{S} = \sqrt{\hat{Z}} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} = e^{-i\pi/4\hat{Z}}$

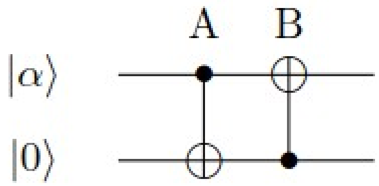
CNOT

$$X \otimes I \to X \otimes X$$
$$I \otimes X \to I \otimes X$$
$$Z \otimes I \to Z \otimes I$$
$$I \otimes Z \to Z \otimes Z$$



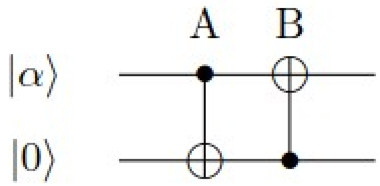
What if some inputs are fixed?

a)



If some inputs are fixed: use stabilizer formalism.
Also track stabilizers S

a)



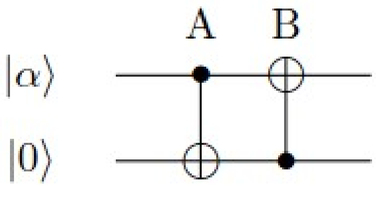
$$S = \{M \in \mathcal{P} \text{ such that } M|\psi\rangle = |\psi\rangle \text{ for all allowed inputs } |\psi\rangle\}$$

Here:
$$S = \{\hat{Z}_2 = \hat{1} \otimes \hat{Z}\}$$
$$\overline{X} = X_1$$
$$\overline{Z} = Z_1$$

Rules for CNOT:

$$X \otimes I \to X \otimes X$$
$$I \otimes X \to I \otimes X$$
$$Z \otimes I \to Z \otimes I$$
$$I \otimes Z \to Z \otimes Z$$

a)



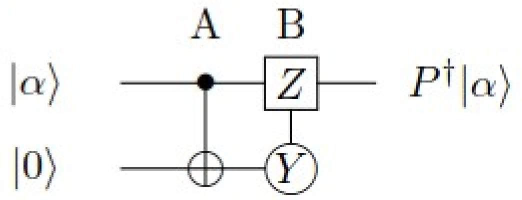
b)

A: CNOT$(1 \to 2)$		$Z \otimes Z$
	$\overline{X}$	$X \otimes X$
	$\overline{Z}$	$Z \otimes I$
B: CNOT$(2 \to 1)$		$Z \otimes I$
	$\overline{X}$	$I \otimes X$
	$\overline{Z}$	$Z \otimes Z$

Figure 4: Alice's improved quantum computer: a) network, b) analysis.

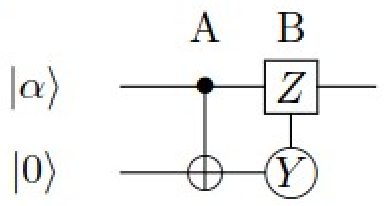
What if some qubits are measured?

a)



If some qubits are measured, update the list of stabilizers

a)



After measurement of Pauli string A:

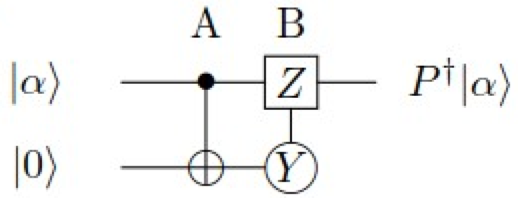
1. Identify $M \in S$ satisfying $\{M, A\} = 0$.
2. Remove M from the stabilizer
3. Add A to the stabilizer
4. For each N, where N runs over the other generators of S and the $\overline{X}$ and $\overline{Z}$ operators, leave N alone if $[N, A] = 0$, and replace N with MN if $\{N, A\} = 0$.

works because every two Pauli strings
either commute or anticommute

This circuit performs the gate

$$\hat{S}^\dagger = \sqrt{\hat{Z}} = \begin{pmatrix} 1 & 0 \\ 0 & -i \end{pmatrix} = e^{i\pi/4\hat{Z}}$$

a)



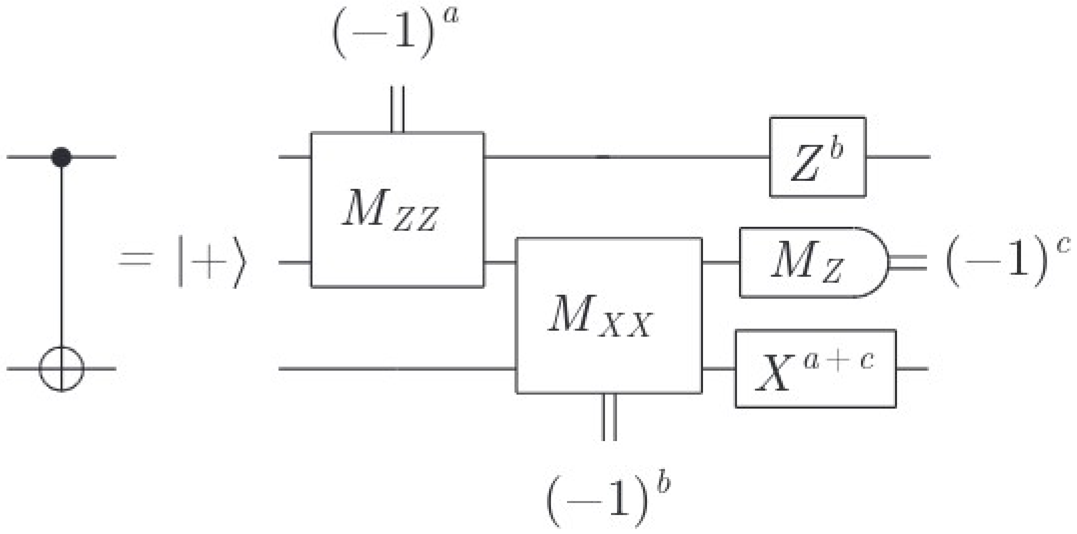
b)

Start		$I \otimes Z$
	$\overline{X}$	$X \otimes I$
	$\overline{Z}$	$Z \otimes I$
A: CNOT$(1 \to 2)$		$Z \otimes Z$
	$\overline{X}$	$X \otimes X$
	$\overline{Z}$	$Z \otimes I$
B: Measure $I \otimes Y$		$I \otimes Y$
	$\overline{X}$	$-Y \otimes Y$
	$\overline{Z}$	$Z \otimes I$

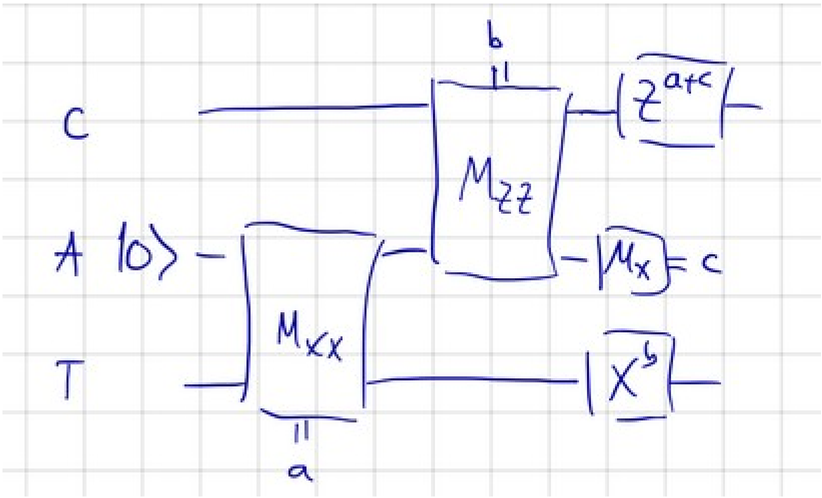
Figure 7: Creating the P gate: a) network, b) analysis.

Old name for S gate

Now we can try to prove



Homework: What about this circuit?



Summary of Gottesman's stabilizer formalism

Begin: Logical inputs = $\overline{X}$ and $\overline{Z}$ operators
Ancillas |0> = Z stabilizers

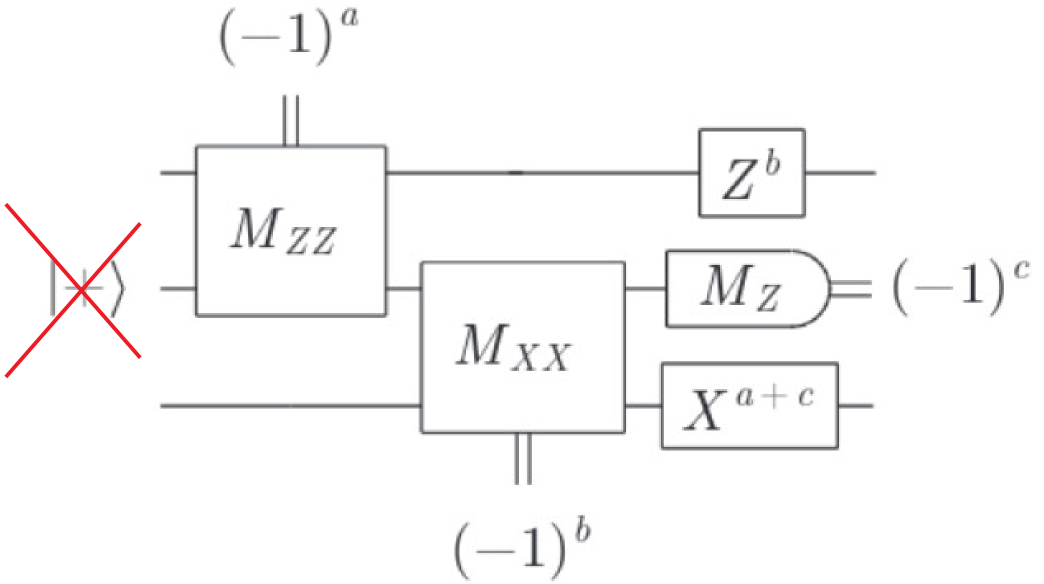
Unitary gates: Update $\overline{X}$, $\overline{Z}$, and stabilizers

Multiqubit Pauli string A measurement: If measurement result random 50%: Update $\overline{X}$, $\overline{Z}$, and stabilizers

1. Identify $M \in S$ satisfying $\{M, A\} = 0$.
2. Remove M from the stabilizer
3. Add A to the stabilizer
4. For each N, where N runs over the other generators of S and the $\overline{X}$ and $\overline{Z}$ operators, leave N alone if $[N, A] = 0$, and replace N with MN if $\{N, A\} = 0$.

If measurement result certain 100%: find which stabilizers make up A to obtain the result. = inverting a matrix, cost n^3

What about measurements whose result is not 50% random?



mapping 3 qubits → 2 qubits,
acquire 3 bits of information,
first measurement not 50%

Aaronson & Gottesman, PRA 2004: “CHP simulator” propagates stabilizer states

Like what we had before, but only stabilizers, no tracking of $\overline{X}, \overline{Z}$

→ find out what a stabilizer circuit does when applied to all |0> input

Decrease cost of numerics for 100% certain measurements, by also storing “destabilizer generators”

The algorithm represents a state by a *tableau* consisting of binary variables x_{ij}, z_{ij} for all $i \in \{1, \ldots, 2n\}$, $j \in \{1, \ldots, n\}$, and r_i for all $i \in \{1, \ldots, 2n\}$ [41]:

$$\left(\begin{array}{ccc|ccc|c} x_{11} & \cdots & x_{1n} & z_{11} & \cdots & z_{1n} & r_1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n1} & \cdots & x_{nn} & z_{n1} & \cdots & z_{nn} & r_n \\ \hline x_{(n+1)1} & \cdots & x_{(n+1)n} & z_{(n+1)1} & \cdots & z_{(n+1)n} & r_{n+1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{(2n)1} & \cdots & x_{(2n)n} & z_{(2n)1} & \cdots & z_{(2n)n} & r_{2n} \end{array}\right)$$

Rows 1 to n of the tableau represent the destabilizer generators $R_1, \ldots, R_n$, and rows $n+1$ to $2n$ represent the stabilizer generators $R_{n+1}, \ldots, R_{2n}$.

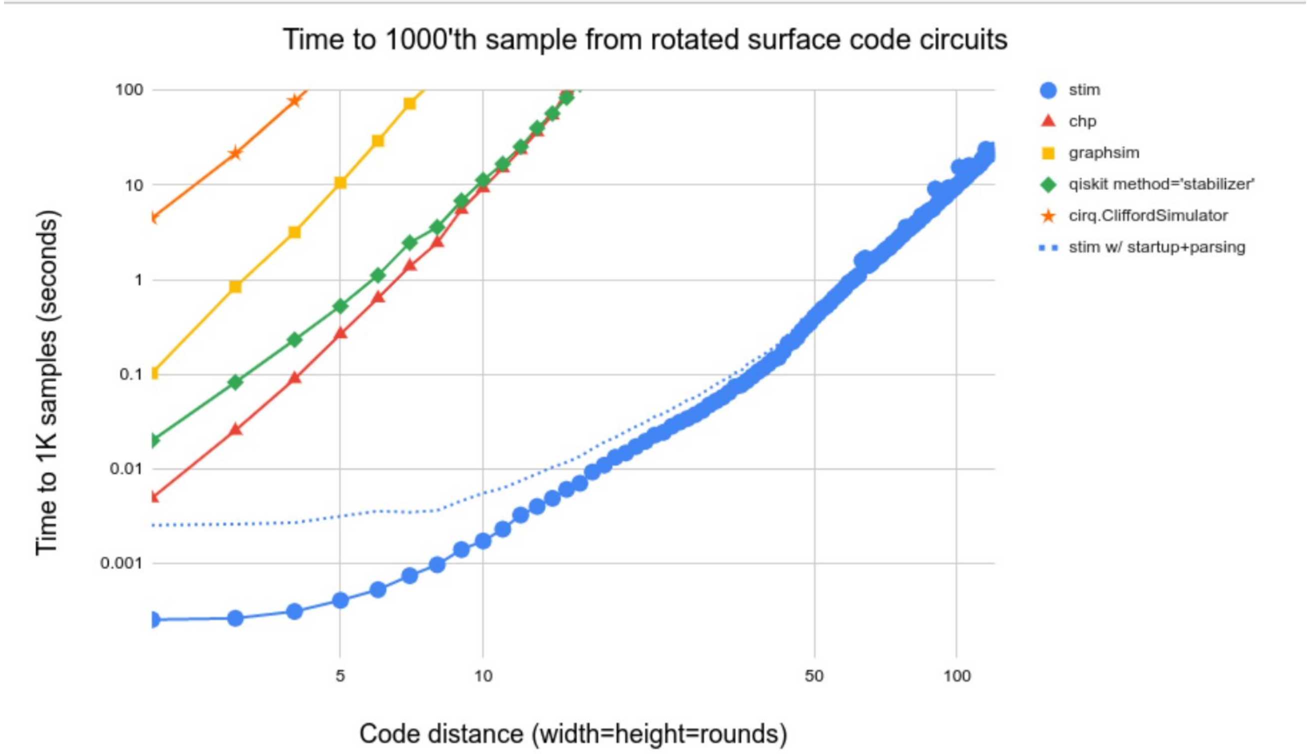
As an example, the 2-qubit state $|00\rangle$ is stabilized by the Pauli operators $+ZI$ and $+IZ$, so a possible tableau for $|00\rangle$ is

$$\left(\begin{array}{cc|cc|c} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right)$$

2021, Craig Gidney (Google): STIM, a Faster tableau Clifford simulator (also storing inverses)

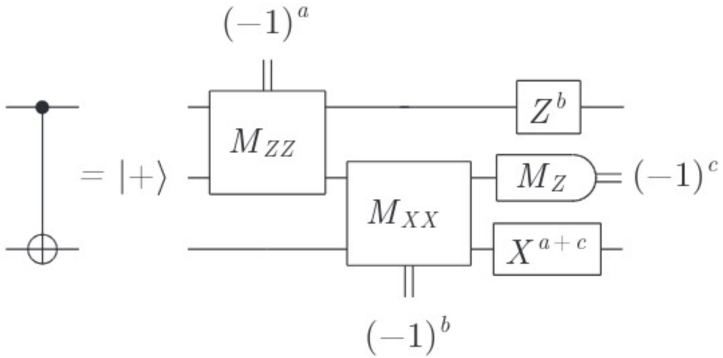
This paper presents "Stim", a fast simulator for quantum stabilizer circuits. The paper explains how Stim works and compares it to existing tools. With no foreknowledge, Stim can analyze a distance 100 surface code circuit (20 thousand qubits, 8 million gates, 1 million measurements) in 15 seconds and then begin sampling full circuit shots at a rate of 1 kHz. Stim uses a stabilizer tableau representation, similar to Aaronson and Gottesman's CHP simulator, but with three main improvements. First, Stim improves the asymptotic complexity of deterministic measurement from quadratic to linear by tracking the *inverse* of the circuit's stabilizer tableau. Second, Stim improves the constant factors of the algorithm by using a cache-friendly data layout and 256 bit wide SIMD instructions. Third, Stim only uses expensive stabilizer tableau simulation to create an initial reference sample. Further samples are collected in bulk by using that sample as a reference for batches of Pauli frames propagating through the circuit.

Today, STIM is the tool of choice for Clifford simulation (important use case: error correction)

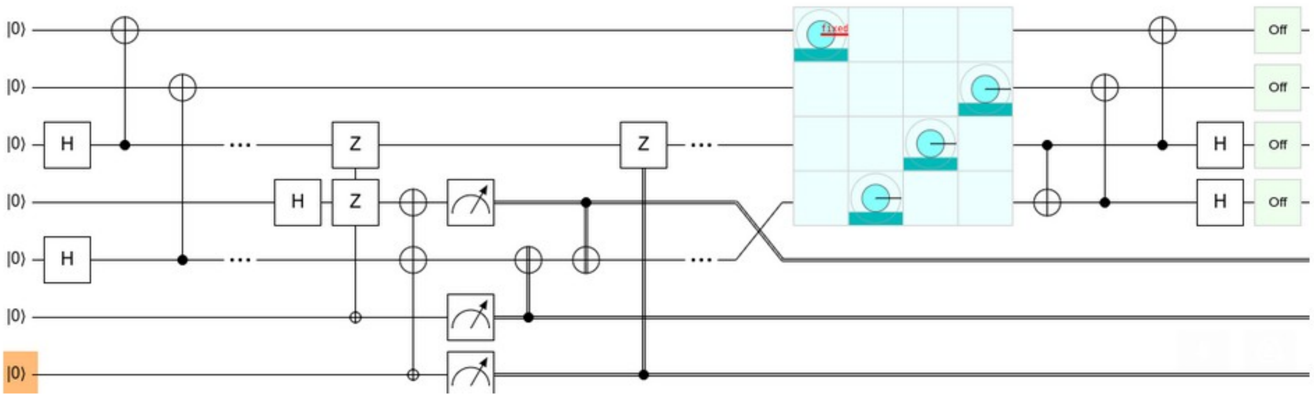


One way to simplify circuits using the improved stabilizer formalism uses state-channel duality

Want to check this:



Response from Craig Gidney (Google, Algassert):



Stabilizer formalism, summary

Typical quantum computation question: from all $|0\rangle$ inputs, apply sequence of gates, what is the measured output statistics?

→ CHP: If all gates Clifford (e.g., CNOT, Hadamard, S phase)
 – simple to simulate, even if highly entangled (CHP algorithm)

Different question: what does a sequence of Clifford gates do as a transformation?

→ Gottesman can be used sometimes