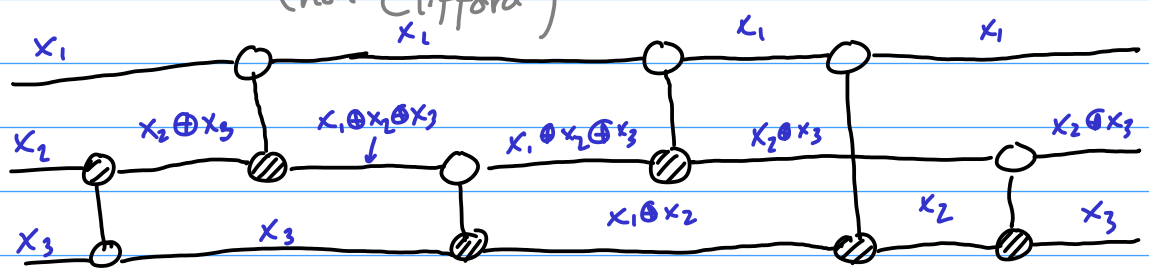
CNOT + phase circuits

(non-Clifford)

$U =$

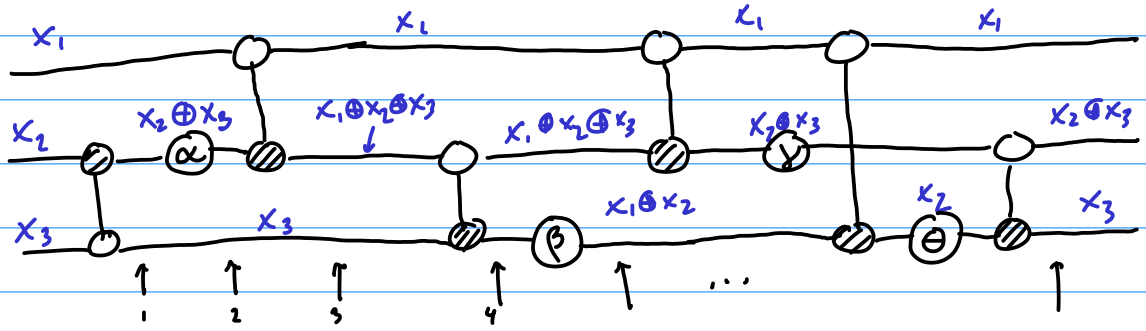


$$U|x_1 x_2 x_3\rangle = |x_1, x_2 \oplus x_3, x_3\rangle$$

Q: What happens when we add phase gates?

$$Z[\alpha] :: |x\rangle \mapsto e^{i\alpha \cdot x}|x\rangle$$

$V =$



$$|x_1 x_2 x_3\rangle \overset{1}{\mapsto} |x_1, x_2 \oplus x_3, x_3\rangle$$

$$\overset{2}{\mapsto} e^{i\alpha \cdot (x_2 \oplus x_3)} |x_1, x_2 \oplus x_3, x_3\rangle$$

$$\overset{3}{\mapsto} e^{i\alpha (x_2 \oplus x_3)} |x_1, x_1 \oplus x_2 \oplus x_3, x_3\rangle$$

$$\overset{4}{\mapsto} e^{i\alpha \cdot (x_2 \oplus x_3)} |x_1, x_1 \oplus x_2 \oplus x_3, x_1 \oplus x_2\rangle$$

$$\mapsto e^{i[\alpha \cdot (x_2 \oplus x_3) + \beta \cdot (x_1 \oplus x_2)]} |x_1, x_1 \oplus x_2 \oplus x_3, x_1 \oplus x_2\rangle$$

$$\mapsto \dots$$

$$\mapsto e^{i[\alpha \cdot (x_2 \oplus x_3) + \beta \cdot (x_1 \oplus x_2) + \gamma \cdot (x_2 \oplus x_3) + \theta \cdot x_2]} |x_1, x_2 \oplus x_3, x_3\rangle$$

Prop Any CNOT+phase circuit describes a unitary of the form:

$$U :: |\vec{x}\rangle \mapsto e^{i\phi(\vec{x})}|L\vec{x}\rangle$$

phase polynomial parity matrix.

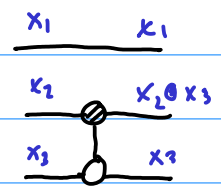
From the example above: $L = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$ and

$$\phi(x_1, x_2, x_3) = (\alpha+\gamma)\cdot(x_2 \oplus x_3) + \beta\cdot(x_1 \oplus x_2) + \theta\cdot x_2$$

phase-folding

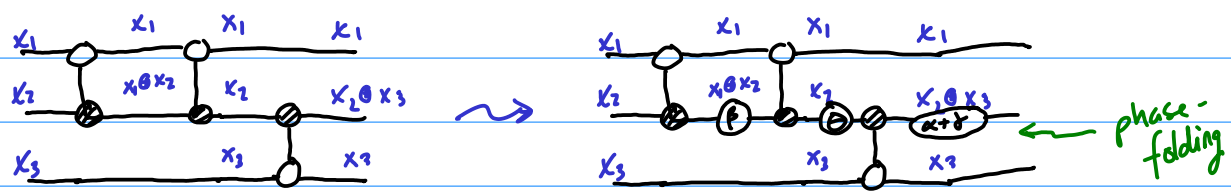
Q: can we re-synthesise a circuit for (L, ϕ)?

For L, we have:



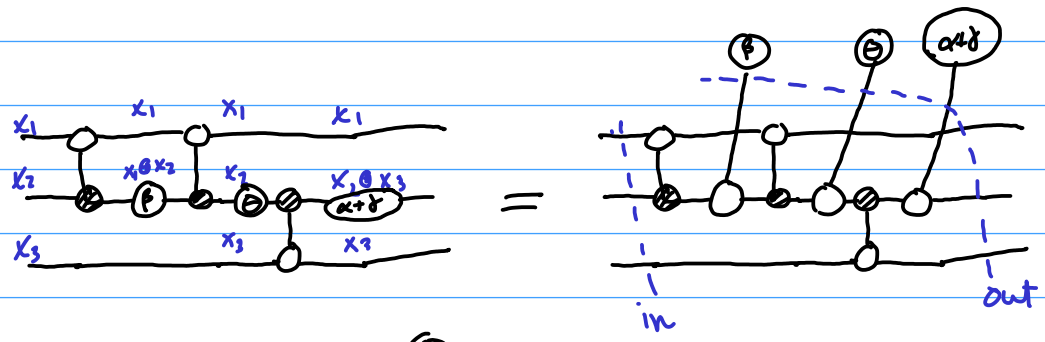
To get ϕ, we need to place Z-phases on wires labelled:
$x_2 \oplus x_3$, $x_1 \oplus x_2$, and x_2

Only $x_1 \oplus x_2$ is missing, so lets (temporarily) create it:

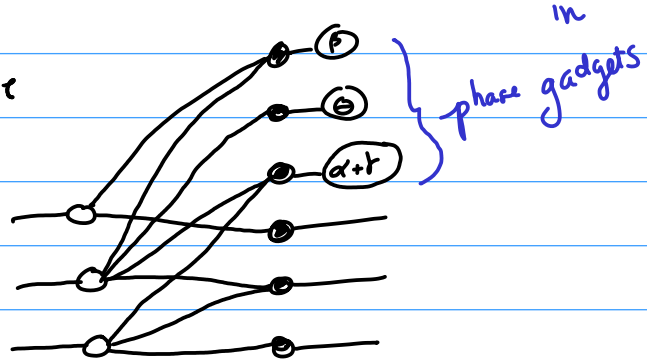


Phase polynomials, graphically (aka. phase gadgets)

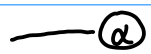
Ex



Phase-free simp



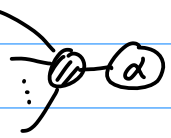
1-legged:



$$:: \ |x\rangle \longmapsto \begin{cases} 1 & \text{if } x=0 \\ e^{i\alpha} & \text{if } x=1 \end{cases} = e^{i\alpha \cdot x}$$

k-legged phase gadget:

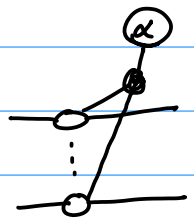
$\sqrt{2}^{(k-1)}$



$$:: \ |x_1 .. x_k\rangle \longmapsto e^{i\alpha \cdot (x_1 \oplus .. \oplus x_k)}$$

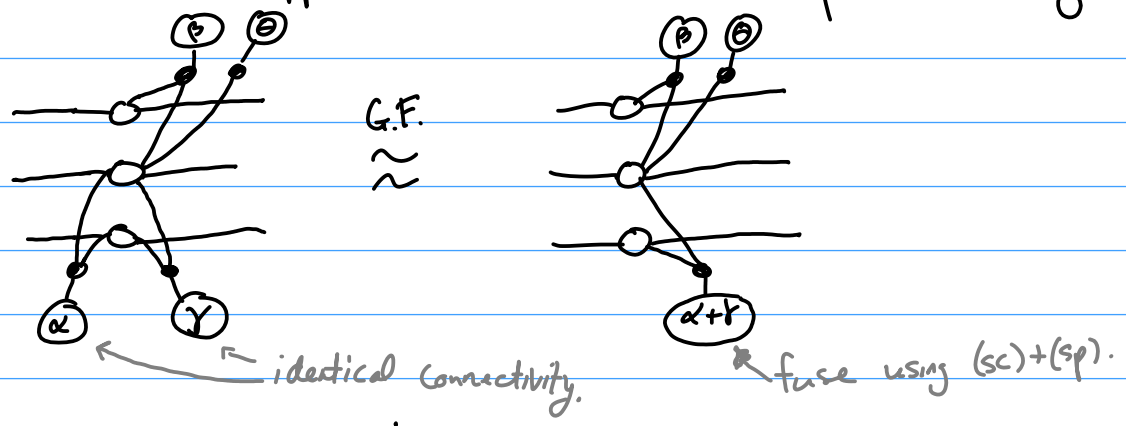
In a diagonal unitary:

$\sqrt{2}^{(k-1)}$

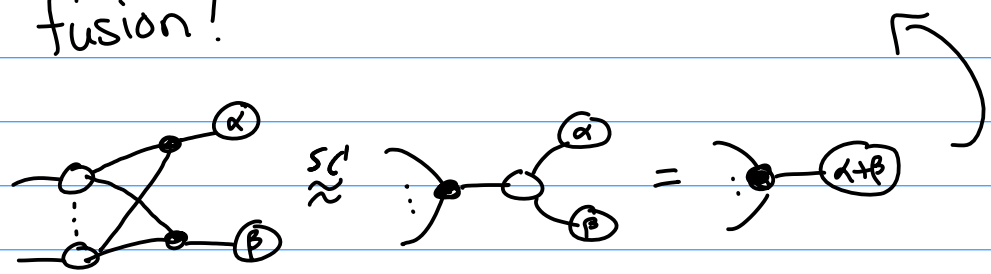


$$:: \ |x_1 .. x_k\rangle \longmapsto e^{i\alpha \cdot (x_1 .. x_k)} |x_1 .. x_k\rangle$$

Q: What happens when there is phase folding?



A: Gadget fusion!



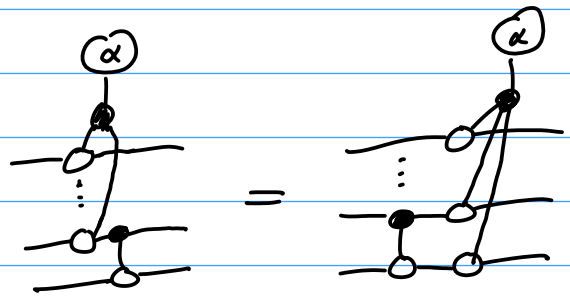
Algorithm: CNOT + phase optimisation. (PNF = Parity NF)

1. unfuse phases and treat as outputs.
2. Compute PNF of phase-free part.
3. perform gadget fusion (* and other phase-poly reductions!)
4. extract a CNOT + phase circuit. ?? →

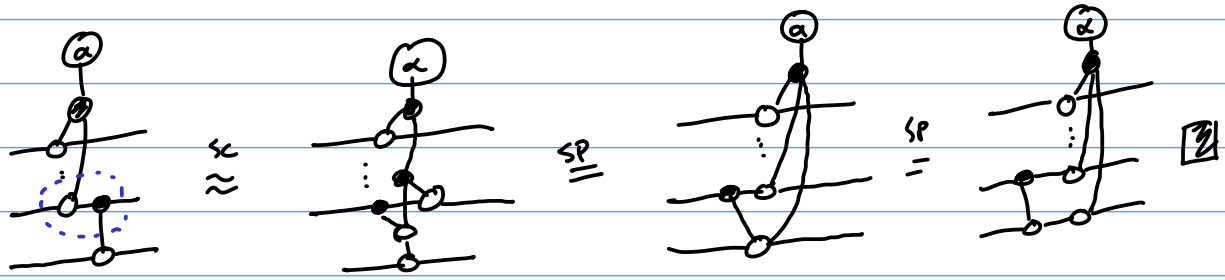
There are choices for step 4.

Naïve approach: "CNOT ladders"

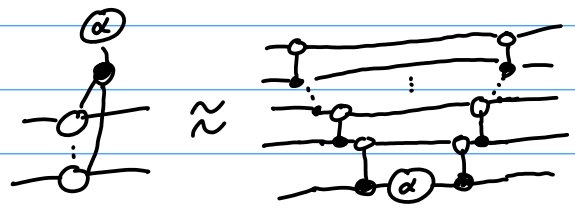
Prop



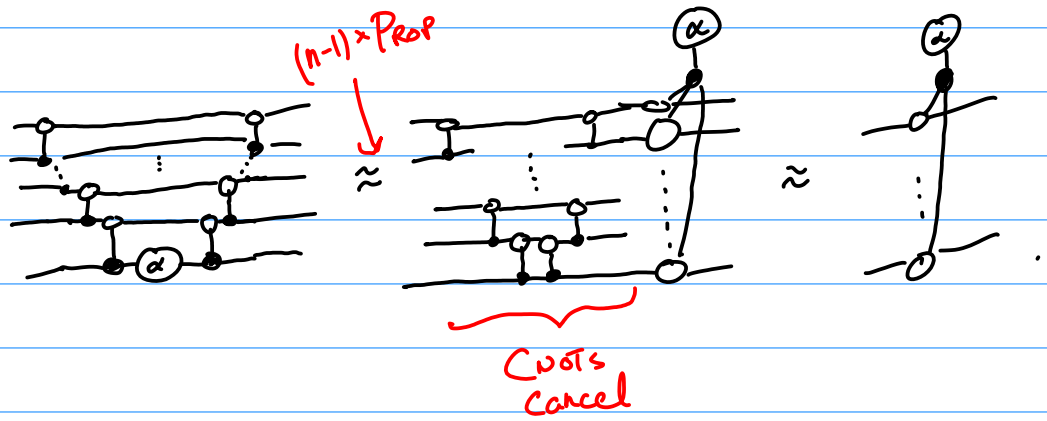
Pf



Cor

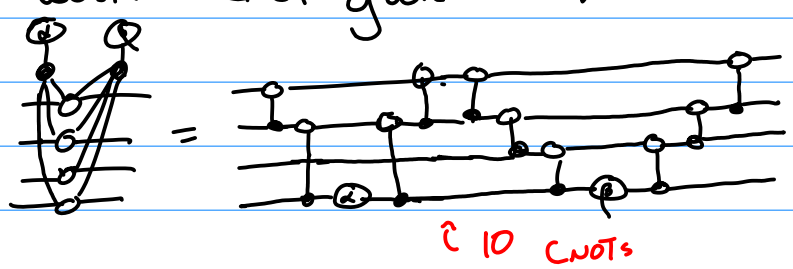


Pf

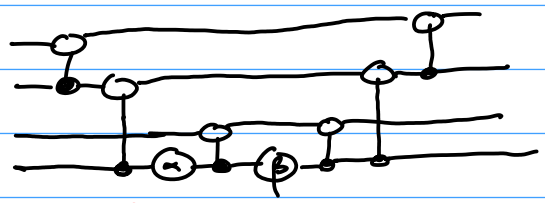


Naïve extraction:
1. unfuse a phase gadget * replace using Cor 1.
2. repeat until no phase gadgets
3. synthesise CNOT circuit from phase-free diag.

* Lots of wasted CNOT gates! e.g.



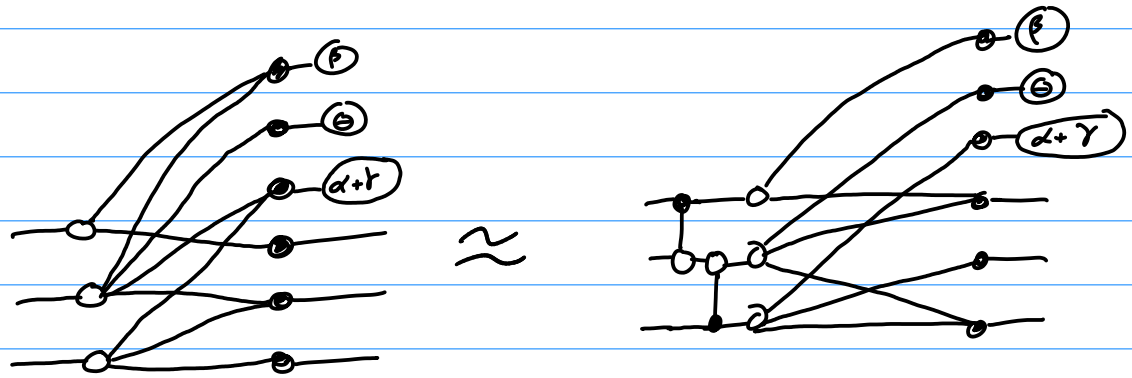
vs.



6 CNOTs

Better extraction

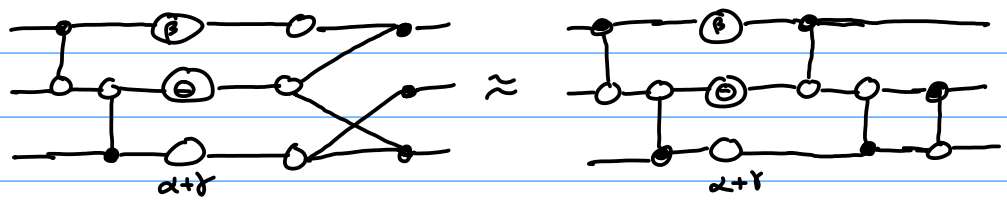
1. write an "extended biadjacency matrix"
2. identify a set of k linearly independent rows
3. reduce each row to a unit vector with column ops.
4. "extract phases" and repeat.



$\overset{SF}{\underset{10}{=}}$ $*$

$$\begin{array}{l} \text{gadgets} \\ \\ \\ \text{outputs} \\ \\ \\ \end{array} \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ \hline 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \xrightarrow{C_2 = C_2 + C_1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \xrightarrow{C_2 = C_2 + C_3} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \hline 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

$*$ =



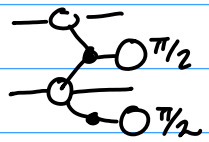
$$\begin{pmatrix} 1 & 1 & 0 \\ \hline 0 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \xrightarrow{C_2 = C_2 + C_1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \xrightarrow{C_2 = C_2 + C_3} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \xrightarrow{C_3 = C_3 + C_2} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

High-level gates

We've seen 2 kinds of phase polynomials:

"Multilinear" form, e.g. 

$:: |x,y\rangle \mapsto e^{i\pi\cdot(-\frac{1}{2}x + x\cdot y)}|x,y\rangle$

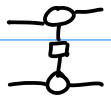
"XOR"/ Phase-gadget form, e.g. 

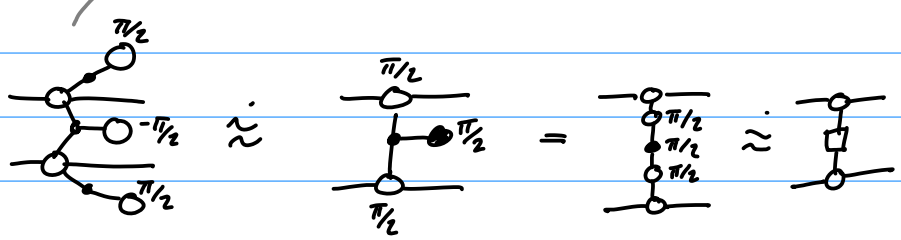
$:: |x,y\rangle \mapsto e^{i\pi\cdot(x\oplus y + y)}$

These two forms are related:

$$\underbrace{x \oplus y}_{\text{XOR}} = \underbrace{x + y}_{\text{plus}} \underbrace{- 2xy}_{\text{"correction"}} \qquad (x,y \in \{0,1\})$$

$$-2xy = x\oplus y - x - y$$
$$\Rightarrow xy = \tfrac{1}{2}(x + y - x\oplus y)$$


$:: |xy\rangle \mapsto e^{i\pi\cdot(xy)}|xy\rangle$
$= e^{i\pi(\frac{1}{2}x + \frac{1}{2}y - \frac{1}{2}x\oplus y)}|xy\rangle$

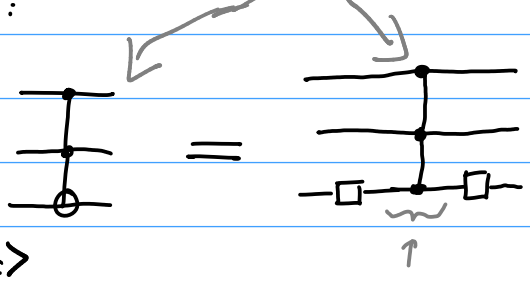


Some gates are easy to write in multilinear form.

Consider :

Q. Circuit Notation

Toffoli:



$|x,y,z\rangle$

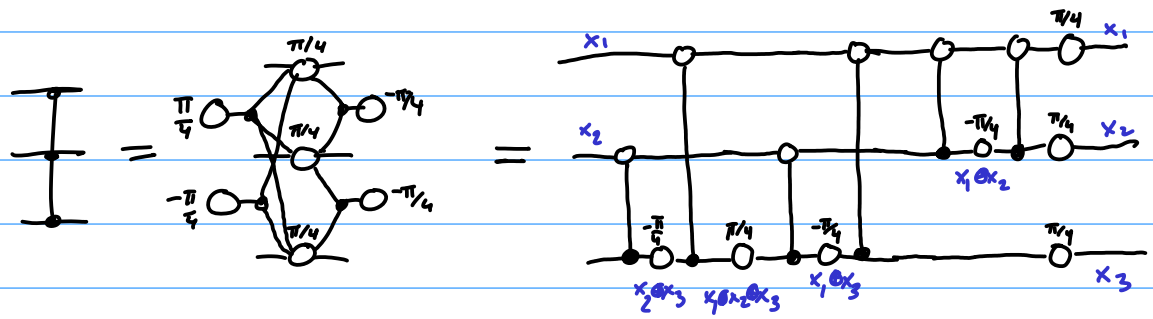
$\mapsto |x,y,(x\cdot y)\oplus z\rangle$

CCZ diagonal

$$CCZ|x_1x_2x_3\rangle = \begin{cases} |x_1x_2x_3\rangle & \text{if } x_1\cdot x_2 = 0 \\ |x_1x_2\rangle \otimes \underbrace{Z|x_3\rangle}_{(-1)^{x_3}|x_3\rangle} & \text{if } x_1\cdot x_2 = 1 \end{cases}$$

$$= (-1)^{x_1x_2x_3}|x_1x_2x_3\rangle = e^{i\pi\cdot x_1x_2x_3}|x_1x_2x_3\rangle$$

$$\begin{aligned} x_1(x_2x_3) &= \tfrac{1}{2}x_1\cdot(x_2+x_3-x_2\oplus x_3) \\ &= \tfrac{1}{2}(x_1x_2+x_1x_3-x_1(x_2\oplus x_3)) \\ &= \tfrac{1}{4}(x_1+x_2-x_1\oplus x_2+\cancel{x_1}+x_3-x_1\oplus x_3-\cancel{x_1}-x_2\oplus x_3+x_1\oplus x_2\oplus x_3) \\ &= \tfrac{1}{4}(x_1+x_2+x_3-x_1\oplus x_2-x_1\oplus x_3-x_2\oplus x_3+x_1\oplus x_2\oplus x_3) \end{aligned}$$



Translation of CCZ into XOR form is a special case of discrete Fourier transform.

Prop For any function $\phi : \mathbb{F}_2^n \to \mathbb{R}$,

dot product, i.e. parities.

$$\phi(\vec{x}) = \frac{-1}{2^{n-1}} \sum_{\vec{y}} \tilde{\alpha}_{\vec{y}} (\vec{x} \cdot \vec{y})$$

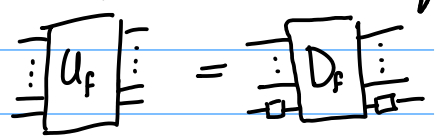
where $\tilde{\alpha}_{\vec{y}} = \frac{-1}{2^{n-1}} \sum_{\vec{z}} (-1)^{\vec{y}\cdot\vec{z}} \cdot \phi(\vec{z})$ are the Fourier coefficients.

In the CCZ case, taking the Fourier xform of $\phi(\vec{x}) = \begin{cases} 1 & \text{if } x_1x_2x_3 = 1 \\ 0 & \text{o.w.} \end{cases}$

gives: $\begin{cases} \tilde{\alpha}_{100} = \tilde{\alpha}_{010} = \tilde{\alpha}_{001} = \frac{1}{4} \\ \tilde{\alpha}_{110} = \tilde{\alpha}_{101} = \tilde{\alpha}_{011} = -\frac{1}{4} \\ \tilde{\alpha}_{111} = \frac{1}{4}. \end{cases}$

This gives a general strategy for synthesising classical oracles $f : \{0,1\}^n \to \{0,1\}$

1. Let $U_f|\vec{x}, y\rangle := |\vec{x}, f(\vec{x}) \oplus y\rangle$



$$D_f|\vec{x}, y\rangle := e^{i\pi \cdot \phi} |\vec{x}, y\rangle \quad \text{where } \phi(\vec{x}, y) = f(\vec{x}) \cdot y$$

2. Compute Fourier coeffs of ϕ.
3. Synthesise D_f as CNOT + Phase circuit.

Pathsums

We know how to deal w/ Cliffords

& w/ CNOT + Phase

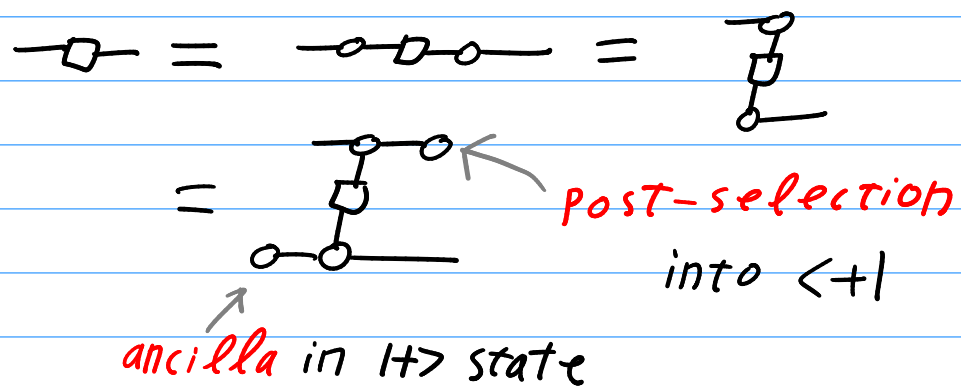
Combination: CNOT + Phase + Hadamard

$\mathbb{Z}[\alpha]$ has $S := \mathbb{Z}[\frac{\pi}{2}]$ as special case

$\Rightarrow$ This is a universal gateset

$\Rightarrow$ Do not expect efficient rewriting

One approach: Pathsums



$|+\rangle = |0\rangle + |1\rangle = \sum_y |+\rangle \qquad \langle +| :: |x\rangle \mapsto 1$

deleting variable

$\text{CNOT} :: |x_1, x_2\rangle \mapsto |x_1, x_1 \oplus x_2\rangle$ updating variable

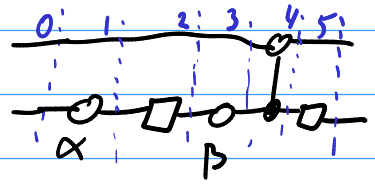
$\mathbb{Z}[\alpha] :: |x\rangle \mapsto e^{i\alpha}|x\rangle$ updating phase polynomial

$\text{-}\square\text{-} :: |x\rangle \mapsto \sum_y (-1)^{x \cdot y} |y\rangle$

y ← Path variable

Ex:

Step 0: $|x_1, x_2\rangle$



$$\overset{1}{\Longrightarrow} e^{i\alpha x_1} |x_1, x_2\rangle$$

$$\overset{2}{\Longrightarrow} \sum_y e^{i\alpha x_1} (-1)^{y \cdot x_1} |x_1, y\rangle$$

$$\overset{3}{\Longrightarrow} \sum_y e^{i\alpha x_1 + \beta y} (-1)^{y \cdot x_1} |x_1, y\rangle$$

$$\overset{4}{\Longrightarrow} \sum_y e^{i\alpha x_1 + \beta y} (-1)^{y \cdot x_1} |x_1, x_1 \oplus y\rangle$$

$$\overset{5}{\Longrightarrow} \sum_{y,z} e^{i\alpha x_1 + \beta y} (-1)^{y \cdot x_1 + (x_1 \oplus y) \cdot z} |x_1, z\rangle$$

Hadamards create new paths, "branches", these interfere via phases

Classical simulation method:
Just sum all the branches

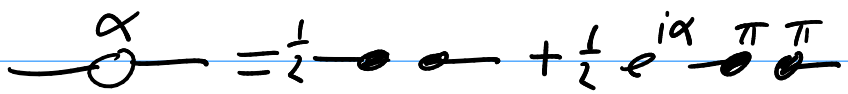
cost: $\mathcal{O}(n \cdot k \cdot 2^h)$ # Hadamards

#qubits #gates

Power of Q. computation = Hadamards?

=> get bunch of 1-qubit circuits

cost: $\mathcal{O}(n \cdot k \cdot 2^c)$ #CNOTs



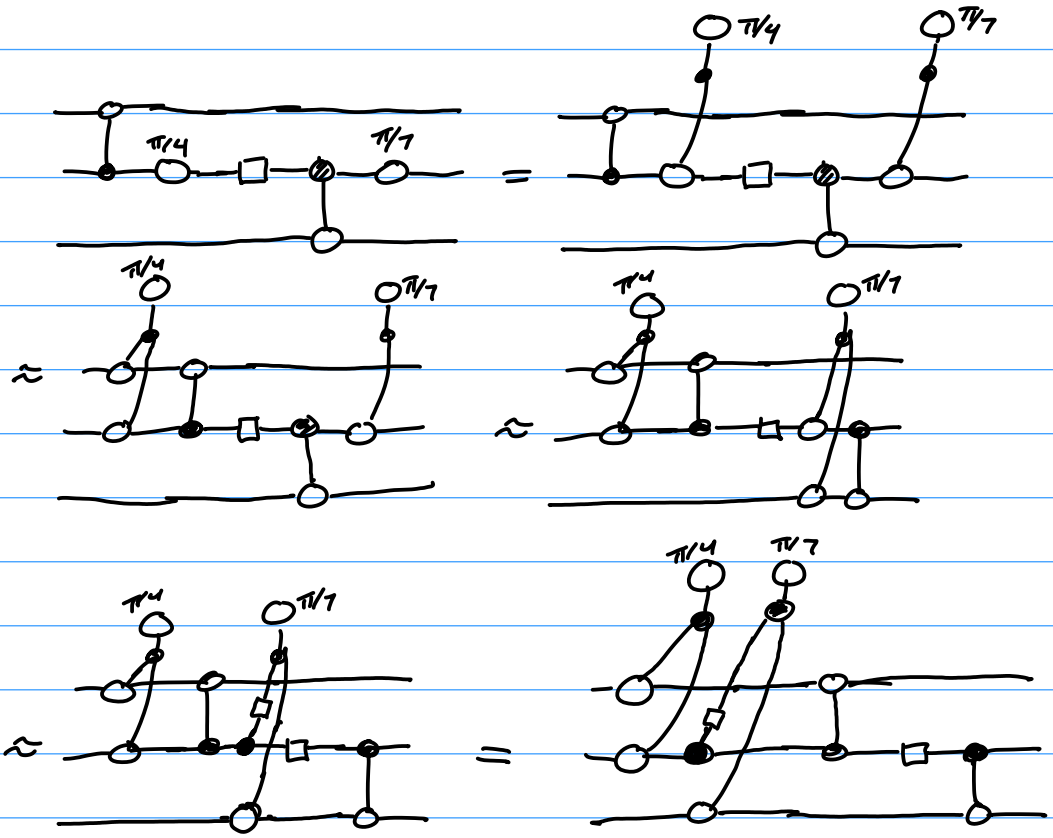
=> get Clifford circuits

cost: $\mathcal{O}(n^2 \cdot k \cdot 2^t)$ # non-Clifford phases

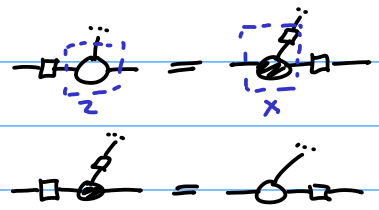
Pauli Gadgets

Clifford + Phase is a universal family.

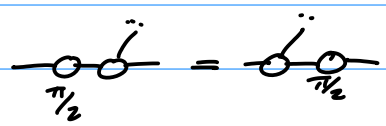
Q: Can we move all the non-Clifford phases out?

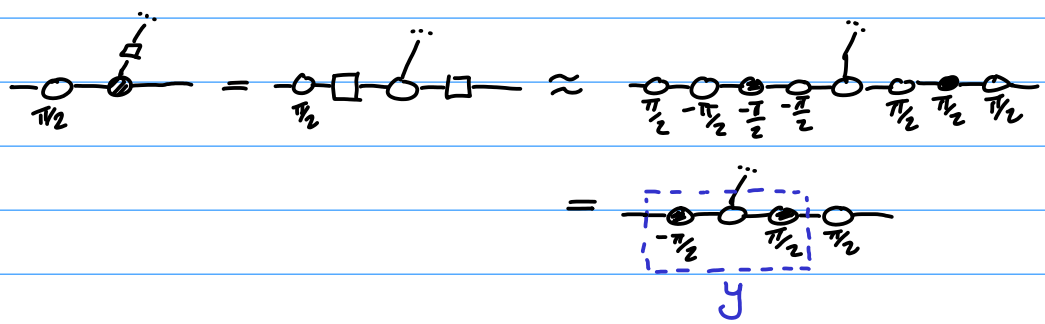


H gates:

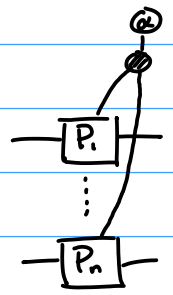


S gates:

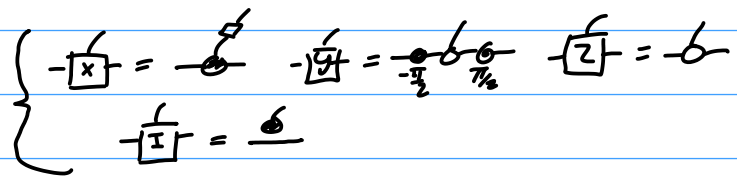




Prop For $\vec{P} = P_1 \otimes \dots \otimes P_n$ with $P_i \in \{I, X, Y, Z\}$ the map:



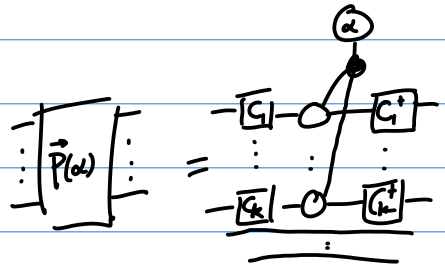
where:



is unitary. It is called the Pauli gadget $\vec{P}(\alpha)$.

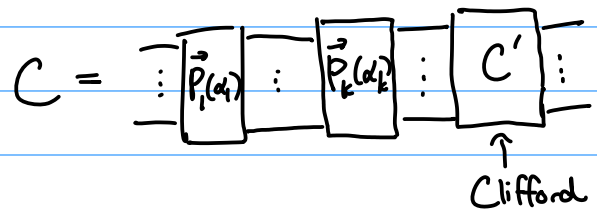
Pf Note

and. So



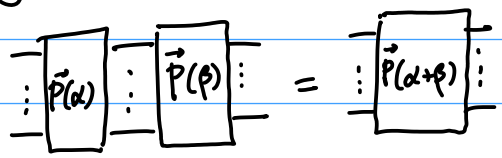
for Cliff. unitaries C_i. Since phase gadgets are unitary, so is $\vec{P}(\alpha)$. ☒

Thm Any Clifford+phase circuit can be written as:

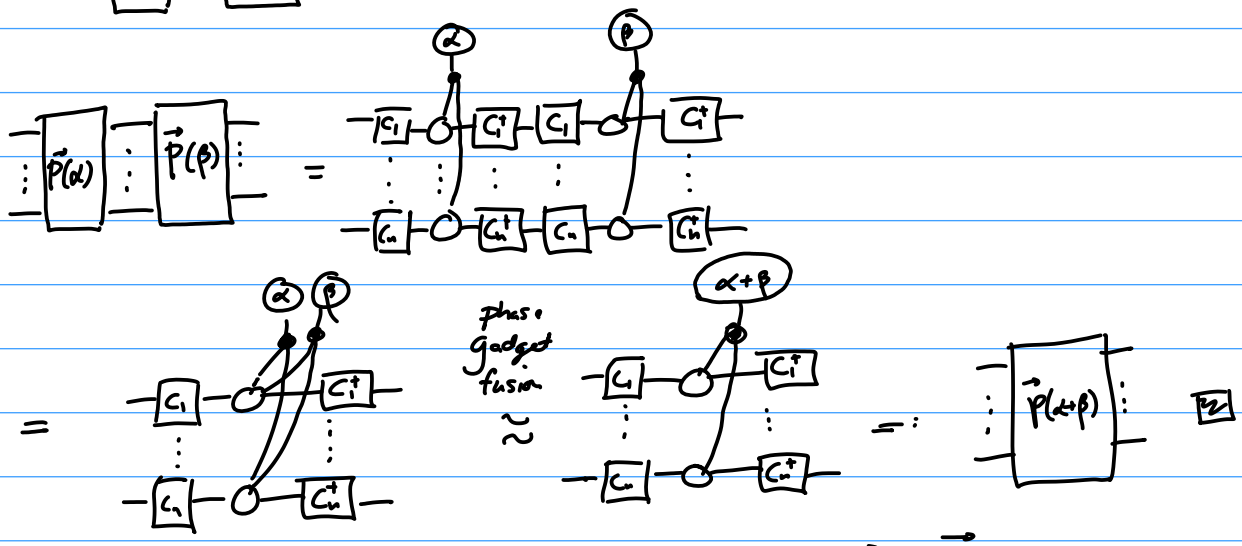


Pf (Idea) • Show Pauli gadgets commute past all Clifford gates.
• Move phases out of C, one at a time. ☒

Prop (Pauli gadget fusion.)



Pf



$\square$

Prop For Paulis $\vec{P}, \vec{Q}$ if $\vec{P}\vec{Q} = \vec{Q}\vec{P}$, then $\vec{P}(\alpha)\vec{Q}(\beta) = \vec{Q}(\beta)\vec{P}(\alpha)$.

Pf Exercise/back (Hint: it's complementarity!)

Algorithm Pauli "phase folding".

1. Compute Pauli gadget form of a circuit.
2. Commute PG's and combine phases where possible.
3. Merge PG's with Clifford phases into the Clifford part.
4. Repeat until no more reductions.
5. Extract circuit.*

* like with CNOT+phase, there are many options.