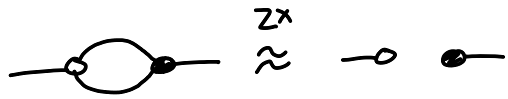
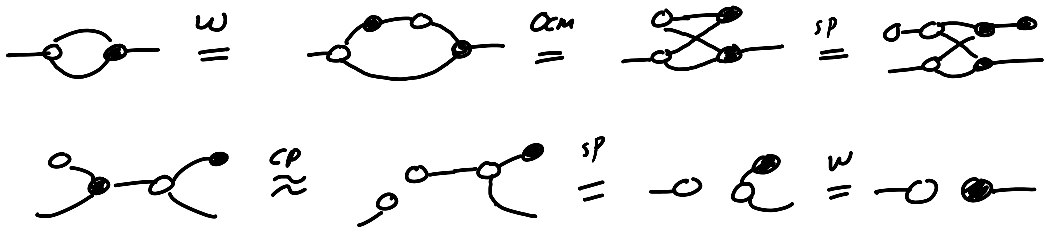
Lecture 9

Rewriting Examples

Thm (COMPLEMENTARITY)

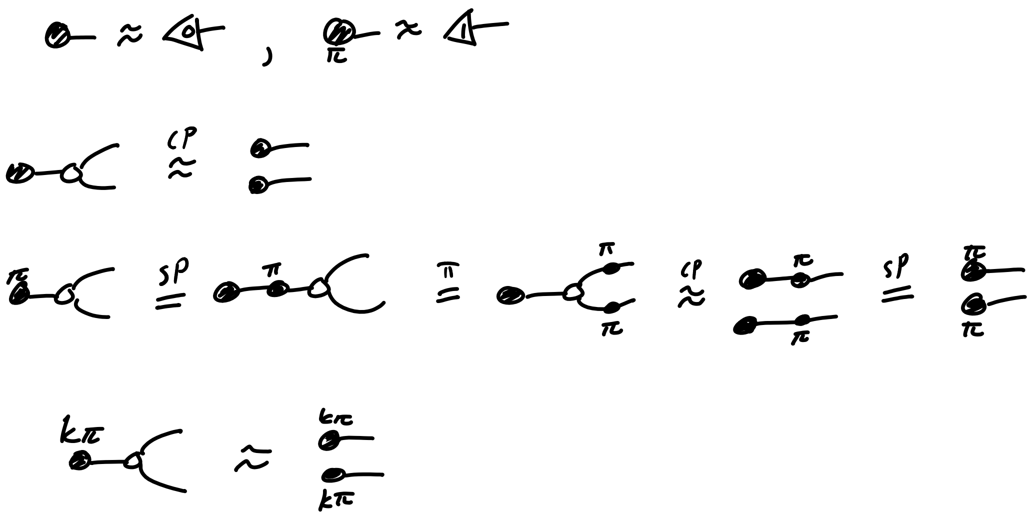


Pf

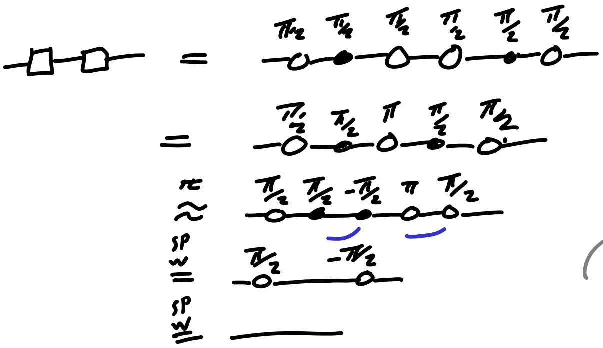


∎

Ex Basis state copy:

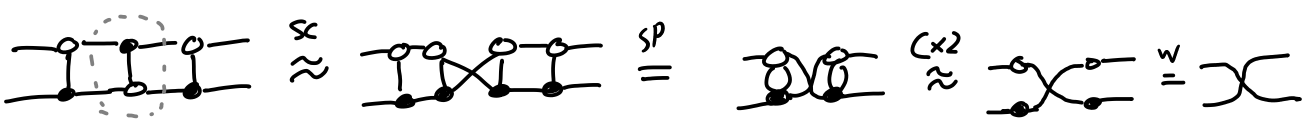


Ex HH



(because $\pi + \frac{\pi}{2} \equiv -\frac{\pi}{2} \pmod{2\pi}$)

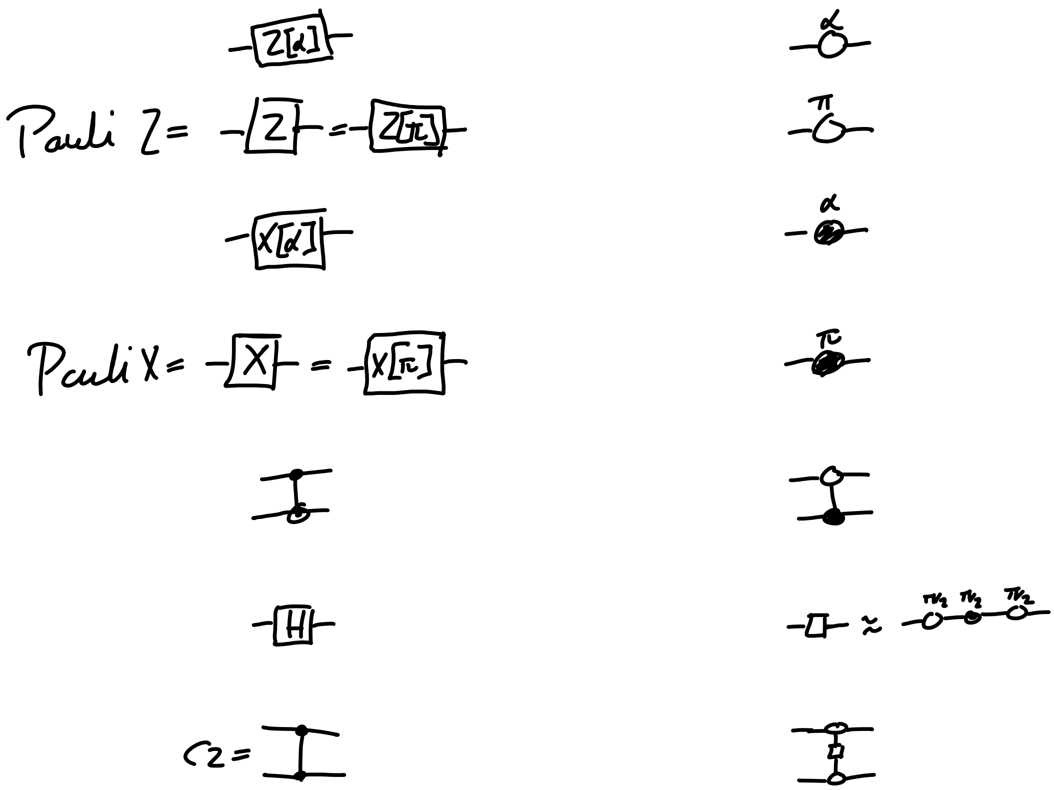
Ex 3CNOT:



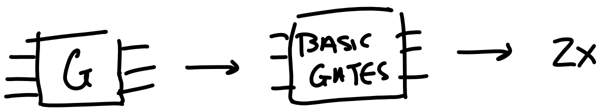
ZX dictionary

CIRCUITS ⟶ ZX-diagrams

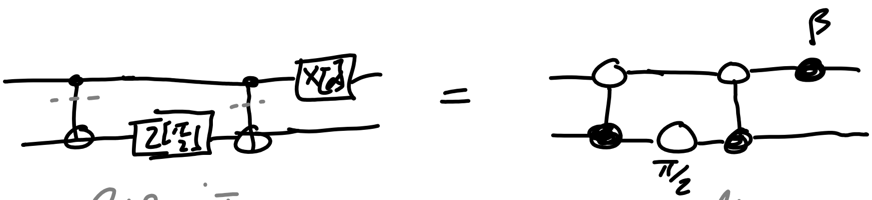
gate | diagr



other stuff
(e.g. CCZ, Tof, ...)



Ex



CIRCUIT

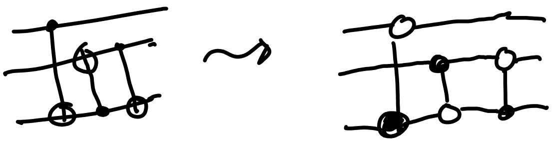
ZX-diag

CNOT Circuits & Phase Free ZX Diagrams

Circuits made just out of 

ZX-Diags made out of  and 

Prop Any CNOT circuit is equal to a phase free ZX-diagram.



Q: What about the converse?

Today: (Unitary) phase-free ZX-diags $\rightsquigarrow$ CNOT circuits.

$$\mathrm{CNOT}\,|x,y\rangle \mapsto |x, x\oplus y\rangle$$

$$\mathrm{CNOT}\,|x,y\rangle \mapsto |f_1(x,y), f_2(x,y)\rangle \quad \text{where} \quad \begin{cases} f_1(x,y) = x \\ f_2(x,y) = x\oplus y. \end{cases}$$

Def A function of the form $f(x_1,\ldots,x_n) = x_{i_1}\oplus\ldots\oplus x_{i_k}$ is called a parity map.

Parities.

<u>Def</u> The field $\mathbb{F}_2$ has elements $\{0,1\}$ where:

$$x \cdot y := x \wedge y \qquad x + y = x \oplus y \qquad (\text{ie. } x+y \bmod 2)$$

Sometimes we call some $x \in \mathbb{F}_2$ a parity.

$$\text{par}(\vec{b}) = \overset{\swarrow \text{ in } \mathbb{F}_2}{\sum_i} b_i$$

$\text{par}(\vec{b}) = 0$ means $\vec{b}$ has an <u>even</u> # of <u>1's</u>

$\text{par}(\vec{b}) = 1$ means <u>odd</u> #.

Parities for subsets of bits:

$$\begin{pmatrix} 1 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix} = b_1 \oplus b_3 \oplus b_4$$

Multiple parities at once:

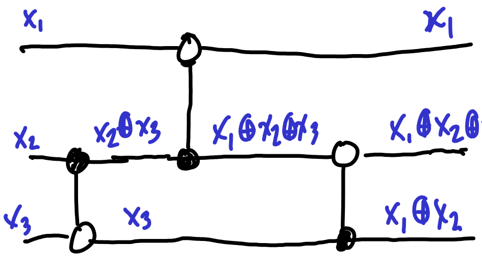
$$\underbrace{\begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{\text{parity matrix.}} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix} = \begin{pmatrix} b_1 \oplus b_3 \oplus b_4 \\ b_2 \oplus b_3 \\ b_1 \oplus b_4 \\ b_4 \end{pmatrix}$$

<u>Thm</u> The action of a CNOT circuit on basis elements is defined by an invertible parity matrix:

$$C|b_1,\dots,b_n\rangle = |c_1,\dots,c_n\rangle$$

where $$P\begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix}.$$

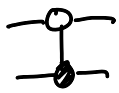
e.g.

$C =$ 

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

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix}}_{P} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_1 \oplus x_2 \oplus x_3 \\ x_1 \oplus x_2 \end{pmatrix}$$

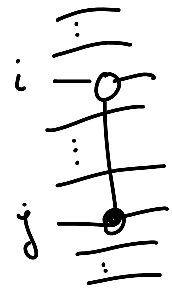
Special case: Single CNOT.



$$|x,y\rangle \mapsto |x, x \oplus y\rangle$$

$$\underbrace{\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}}_{P} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x \\ x \oplus y \end{pmatrix}$$

More generally:



$\longleftrightarrow$ $\begin{array}{c} \\ j \to \end{array} \begin{pmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & \square & & 1 & \\ & & & & 1 \end{pmatrix} = \underbrace{E^{ij}}_{\text{elementary matrix}}$, with $\square$ in column i

$$E^{ij}A = A'$$

A': row j = row j + row i

$$A E^{ji} = A'$$

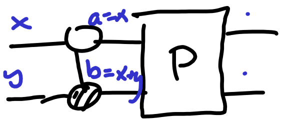
A': col j := col i + col j

Suppose $P E^{i_1 j_1} \cdots\cdots E^{i_k j_k} = I$,

then $P = E^{i_k j_k} \cdots E^{i_1 j_1}$

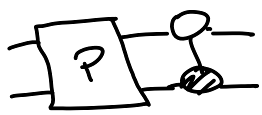
P: parity matrix

$E^{i_k j_k}, \dots, E^{i_1 j_1}$: CNOT gates!



$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} a+b & b \\ c+d & d \end{pmatrix}$$

$C_1 = C_1 + C_2$

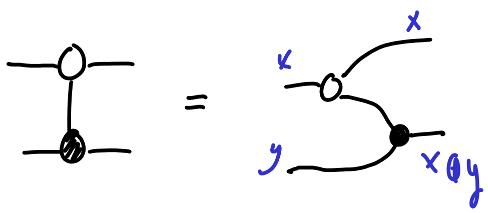


$$\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} a & b \\ a+c & b+d \end{pmatrix}$$

$R_2 = R_2 + R_1$

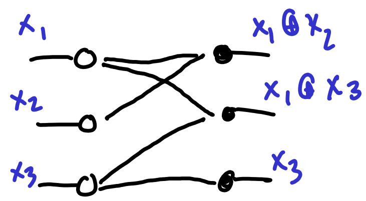
Algorithm: CNOT-SYNTH:

* Start w/ Parity matrix P, empty circ. C.
* Do Gauss-Jordan reduction of columns of P.
  - whenever an elem. col operation E^{ji} is applied, append CNOT^{ji} to C.
* C now implements P.



More general parity maps:

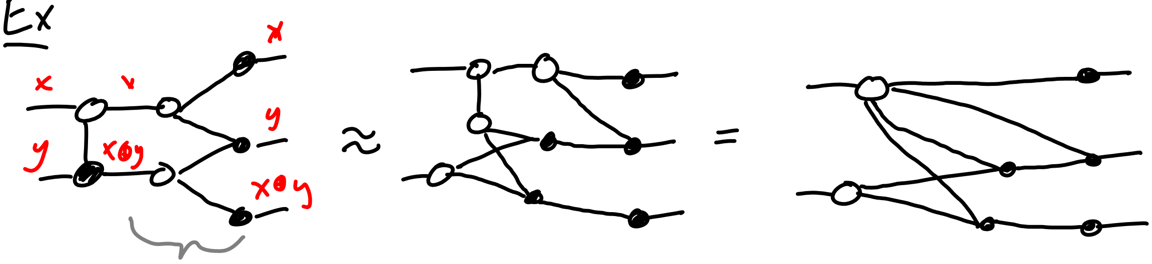
$$P = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} x_1 \oplus x_2 \\ x_1 \oplus x_3 \\ x_3 \end{pmatrix}$$



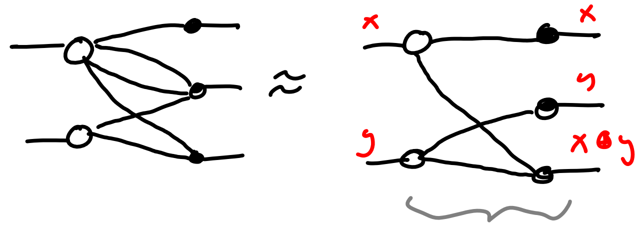
$\longleftarrow$ implements P!

Lecture 10

Ex

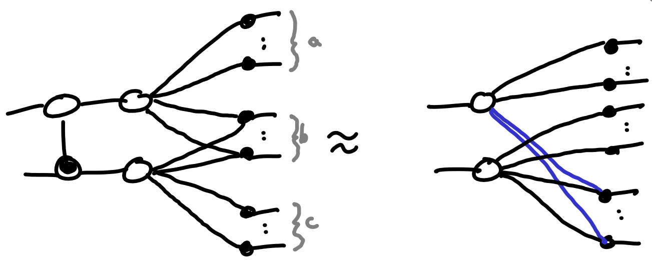


$$\begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix}$$



$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$$

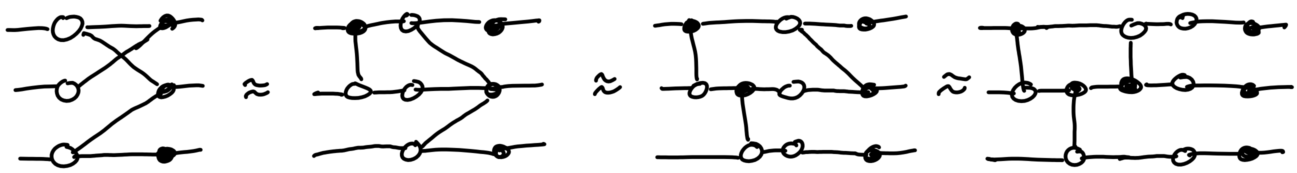
Lem 4.2.3



$$\begin{array}{c} a\{ \\ \\ b\{ \\ \\ c\{ \end{array}\begin{pmatrix} 1 & 0 \\ \vdots & \vdots \\ 1 & 0 \\ 1 & 1 \\ \vdots & \vdots \\ 1 & 1 \\ 0 & 1 \\ \vdots & \vdots \\ 0 & 1 \end{pmatrix} \underset{C_1 = C_1 + C_2}{\overset{C_1 = C_1 + C_2}{\rightleftarrows}} \begin{pmatrix} 1 & 0 \\ \vdots & \vdots \\ 1 & 0 \\ 0 & 1 \\ \vdots & \vdots \\ 0 & 1 \\ 1 & 1 \\ \vdots & \vdots \\ 1 & 1 \end{pmatrix}$$

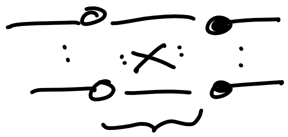
Ex

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



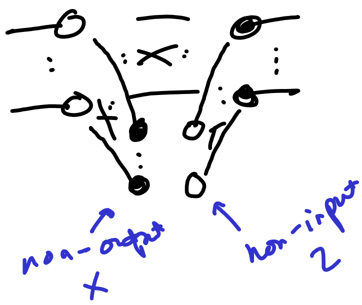
Def A spider is called
* an input spider if it is conn. to an input
* an output spider ... output
* an interior spider otherwise.

Def A phase-free ZX-diagram is in parity normal form
- every Z spider is conn. to exactly 1 input
- every X ... ... output
- no wires between spiders of the same type
- no parallel wires



Def A ZX-diagram is called 2-coloured if there are no Z-Z edges, X-X edges, parallel edges or self-loops.

Def A phase-free ZX diag. is in generalised parity form if it is 2-coloured and:
1. every input is conn. to Z, output to X
2. every sp. is conn to at most 1 input/output
3. no scalar spiders
* 4. no edges between interior spiders



Algorithm 2: Reduction to generalised PNF.

1. apply (sp), (comp), and $\circ = \bullet = 2$ as much as possible.
2. try to apply (sc) to a pair $\circ - \bullet$ where:
   - $\circ$ is not an input
   - $\bullet$ is not an output
3. if step 2 applied (sc), goto step 1. otherwise:
4. Use (id) to make sure every input is conn. to Z & output conn. to X.

Thm Alg 2 terminates efficiently in generalised PNF.

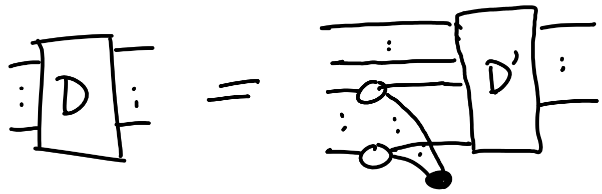
Pf (sketch) • Each iteration of steps 1–3 removes non-input Z spiders (and non-output X-spiders) without making new ones.

$\Rightarrow$ # iterations bounded by # of spiders

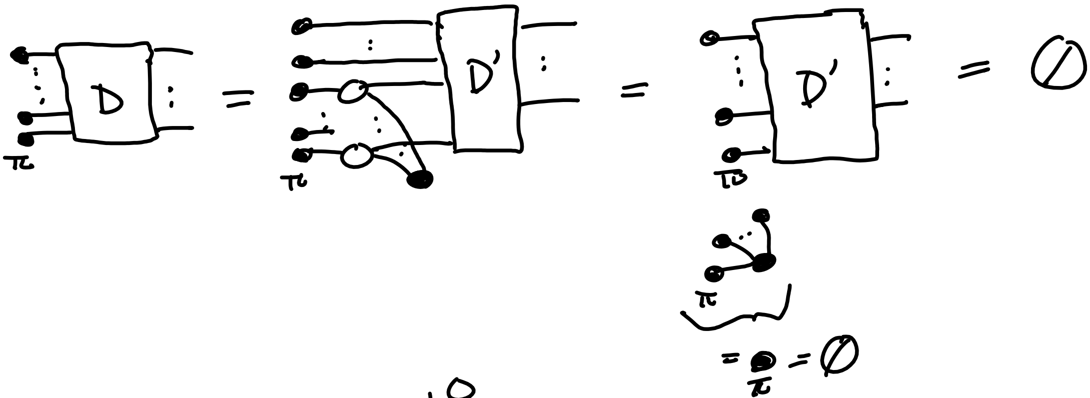
- after the loop, conds 3–5 are satisfied.
- after step 4, conds 1–2 are satisfied. $\boxtimes$

Prop If D is unitary and in generalised PNF, then it is in PNF.

Pf If D has an interior X spider, then:



So:



So, there exists $|\psi\rangle^{\neq 0}$ s.t. $D|\psi\rangle = \emptyset$. But:

$$D^\dagger D|\psi\rangle = |\psi\rangle \neq 0. \quad \lightning$$

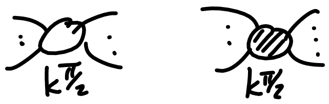
So D has no interior X-spiders. Similarly,

D has no
- Z-sp's connected to >1 input
- interior X sp's
- X-sp's connected to >1 output. $\boxtimes$

$$\text{Unitary Phase-free ZX} \overset{*}{\Longrightarrow} \text{PNF} \Longrightarrow \text{CNOT circuit}.$$

Clifford diagrams and circuits

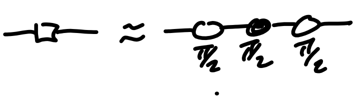
Def A ZX-diagram is Clifford when it is made of Clifford spiders

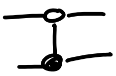


$k \in \mathbb{Z}$.

Def Clifford circuits are circuits made from:

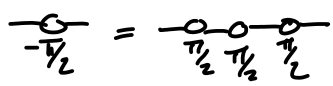
$S =$ 

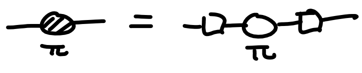
$H =$ 

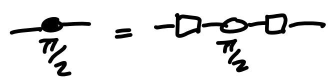
$CNOT =$ 

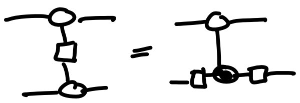
Ex Some common Clifford gates:

$Z =$ 

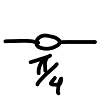
$S^\dagger =$ 

$X =$ 

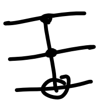
$\sqrt{X} =$ 

$CZ =$ 

Ex Some non-Clifford gates:

$T =$ 

$(S = T^2)$

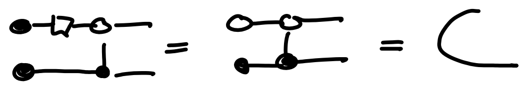
$ToF =$  $= \dots$

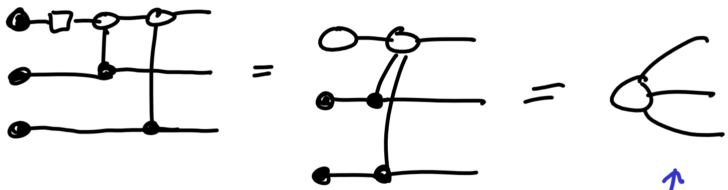
Lecture 11

Def A Clifford state is a state $|\psi\rangle = C|0...0\rangle$ for a Clifford circuit C.

Q: Why care about Cliffords?

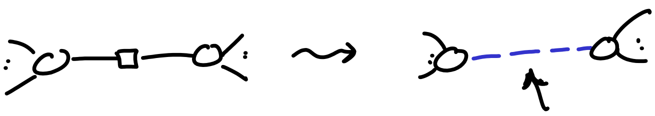
* Contains useful states, e.g.

Bell $\approx$ 

GHZ $\approx$ 

e.g. nonlocality in PQP

* Quantum error correction (later)
* Eff. classical simulation (soon)
* Rich rewrite theory (now!)

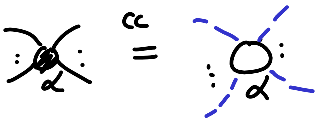
Notation: 

Hadamard edge

Def A ZX diagram is graph like if:

1. all spiders are Z spiders
2. all edges btw spiders are Hadamard edges
3. no parallel edges or self-loops
4. every input/output is connected to a spider.

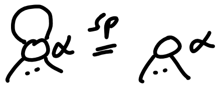
Prop Every ZX-diagram is equal to a graph-like one.

Pf 1. Use  to elim X spiders.

& use  to cancel extra H's.

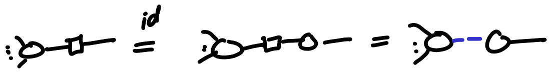
2. Use (sp) to elim non-H edges: 

3. For parallel H-edges:

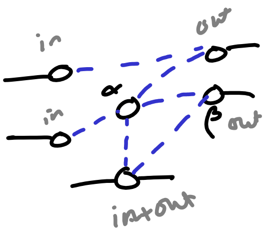
For self-loops: 

4. Use  if necessary.

e.g.  ⟵ g.l.



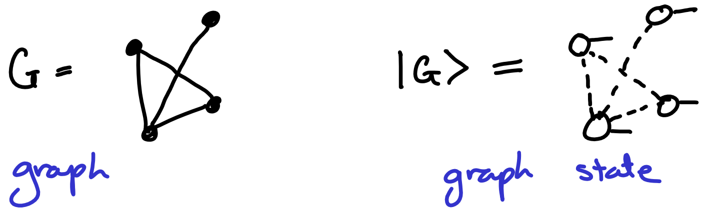
$\boxtimes$

Ex 

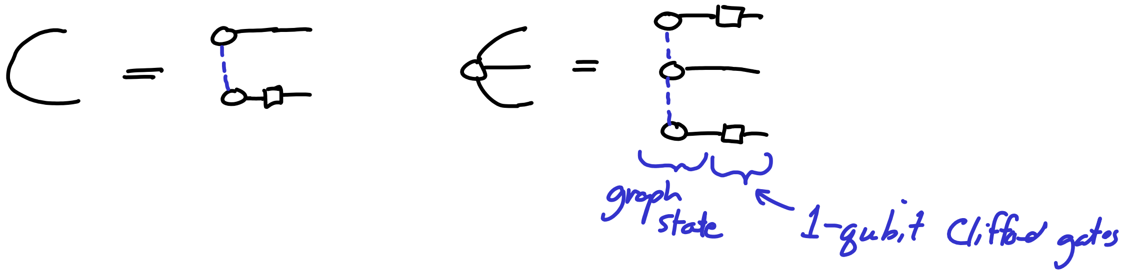
Def A graph-like diagram is called a <u>graph state</u> if:

- no inputs
- no interior spiders
- no phases

Ex



Some states are almost graph states, e.g.

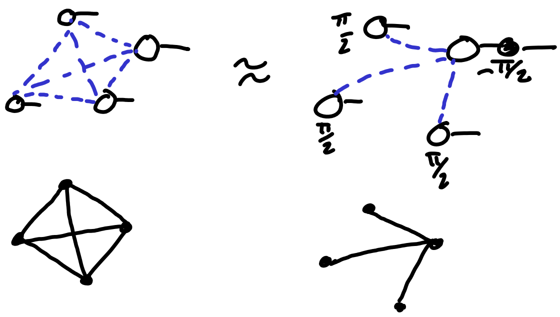


Def A graph state with local Cliffords (GSLC) is a state of the form $(U_1 \otimes \ldots \otimes U_n)|G\rangle$ for some graph state $|G\rangle$ and 1-qubit Clifford gates U_i.

Thm Any Clifford state is = to a GSLC.

We'll need some new tools to prove this!

First, note that for GSLCs, the graph can be deceiving!

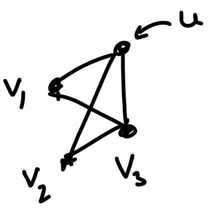


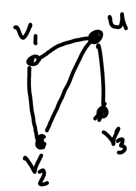
Def Let $G=(V,E)$ be a graph and $u \in V$.

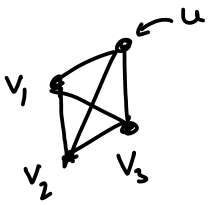
The Local complementation of G about u is a new graph $G*u=(V,E')$ where

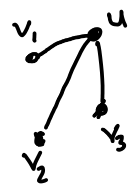
$$\forall v,w \in \underset{\uparrow \text{ neighbourhood}}{N_G(u)} . \quad (v,w)\in E' \iff (v,w)\notin E .$$

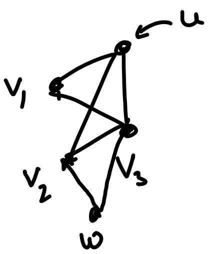
Ex

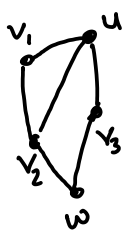
1. $G=$ 

$G*u=$ 

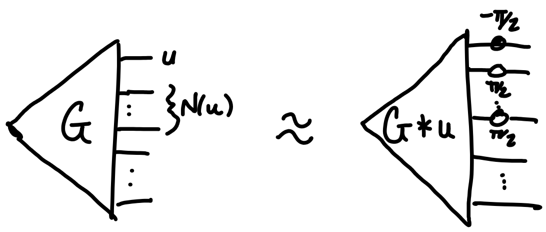
2. $G=$ 

$G*u=$ 

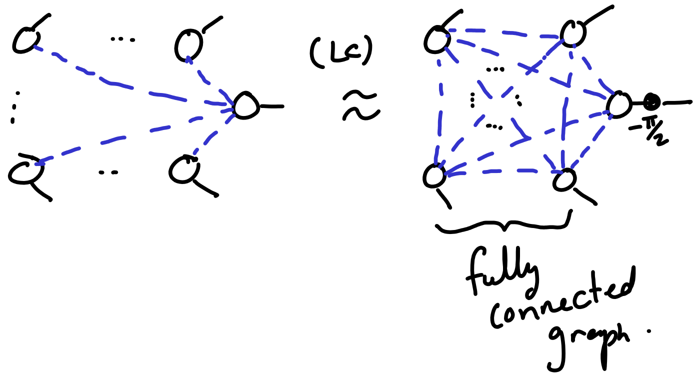
3. $G=$ 

$G*u=$ 

Prop

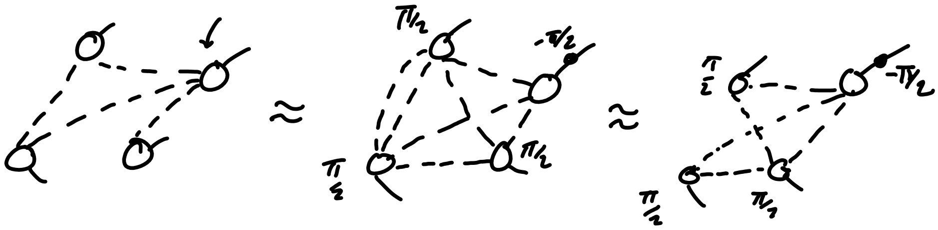


Graphically:

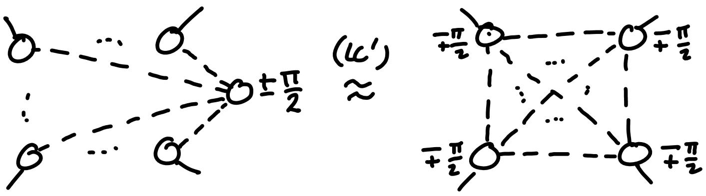


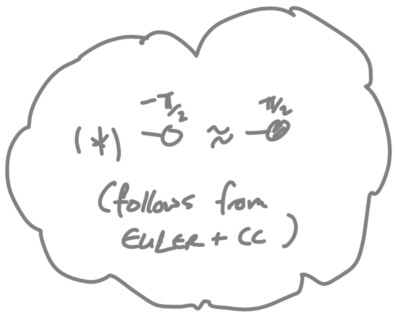
Q: Why is this the same as local comp?

A: Because 

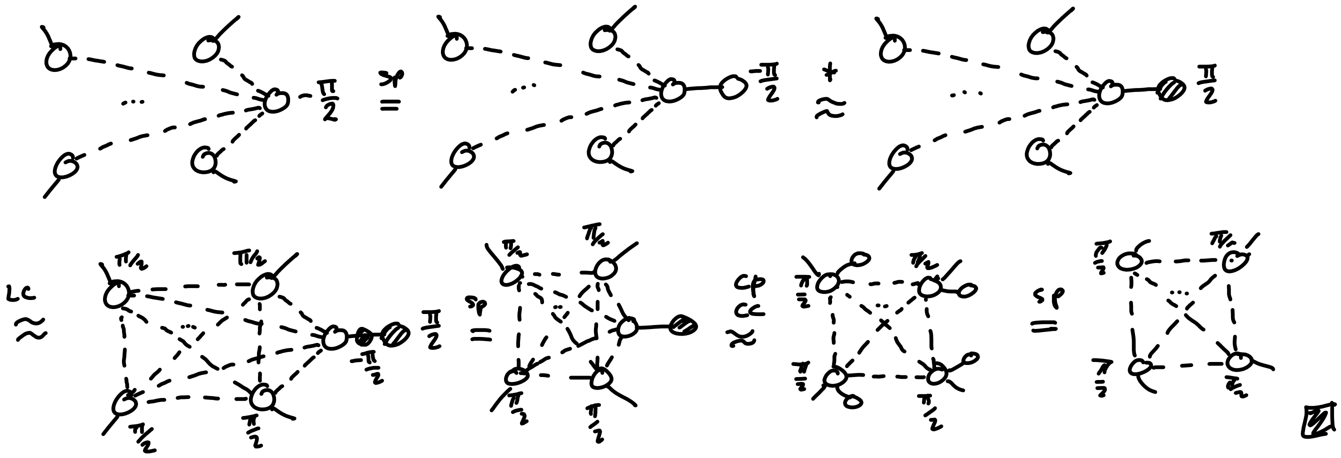


Prop



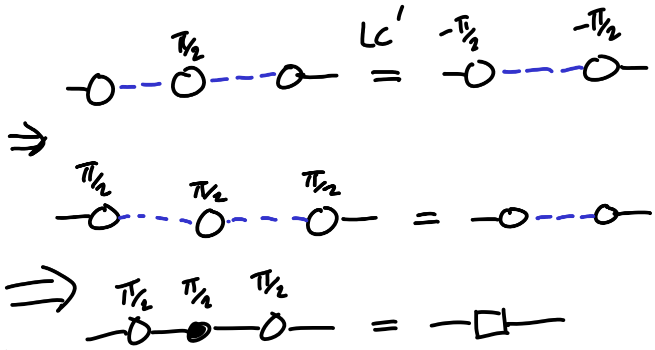


Pf



$\boxtimes$

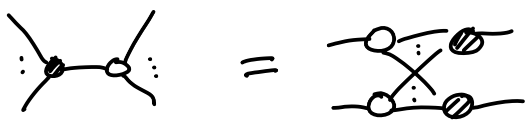
$LC' \Rightarrow EU$



(nb. $EU \Rightarrow LC'$, but harder)

Pivoting.

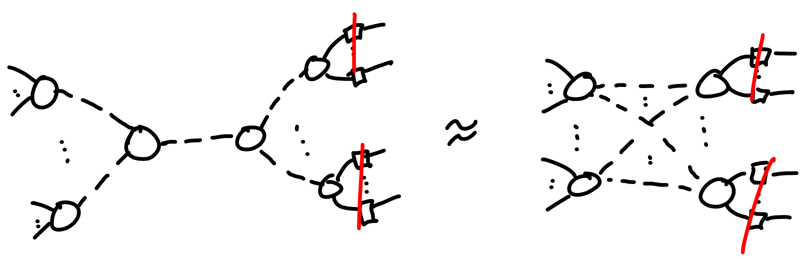
Consider the (sc) rule:

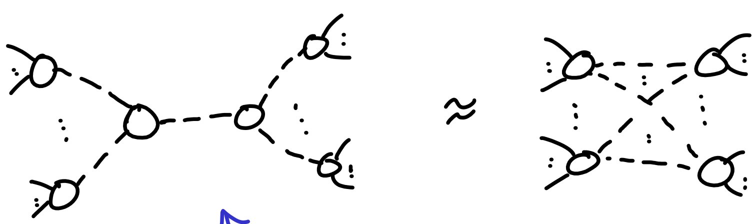


add some context:

always deletes spiders!

Now, (cc) both sides to elim X spiders:

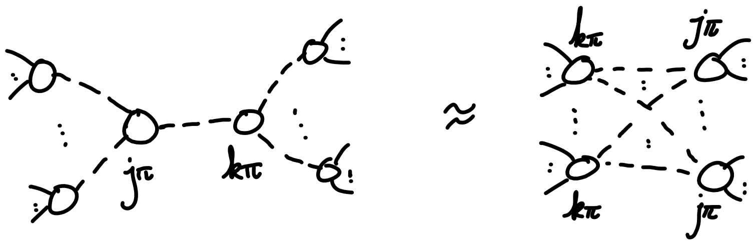




deletes 2 adj. phase-free spiders.

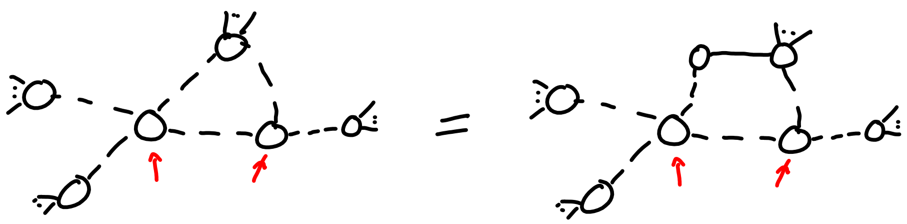
Generalisation:

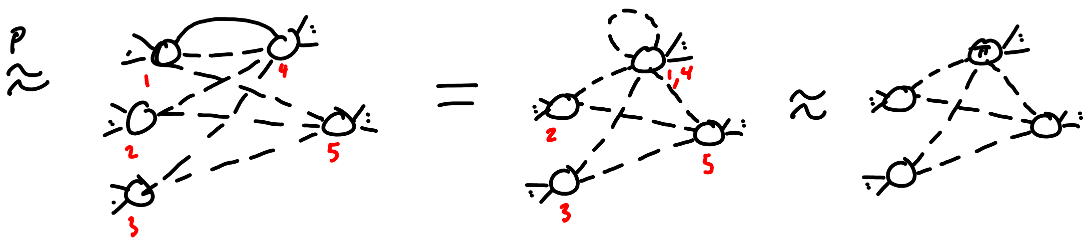
Pivot rule:



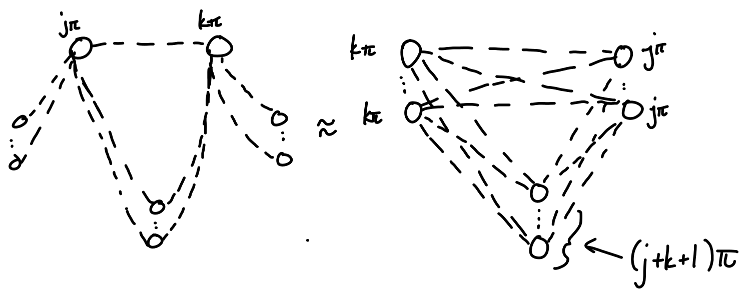
deletes 2 adjacent Pauli spiders.

Q: What if they share neighbours?

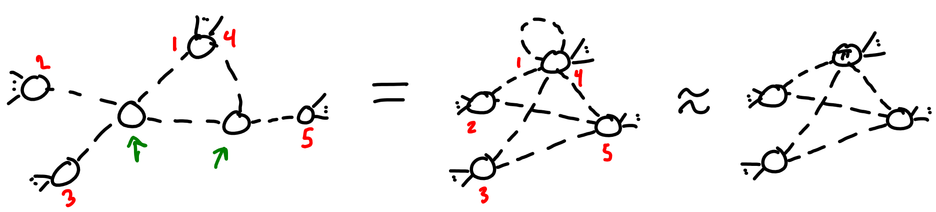




General pivot:



Or, as I prefer to think about it, use the simpler rule, but allow boundary sp's to match twice:



Rewrite strategy (Clifford-simp.)

1. convert to a graph-like diagram
2. apply LC′ & P′ as long a possible.
3. remove isolated $\{0,\pi\}$-spiders.

Prop 1 Clifford-simp terminates for any ZX-diag and removes all interior:

* $\pm\frac{\pi}{2}$ spiders
* pairs of connected $\{0,\pi\}$-spiders

Recall: $m \vdots \boxed{} \vdots\, n$ is a $2^n \times 2^m$ matrix.

$m=n=0 \Rightarrow 2^0 \times 2^0 = 1\times 1$ matrix (a scalar)

Def A scalar ZX-diagram is a ZX-diag w/ no inputs and no outputs.

Cor (to Prop 1) There exists a terminating rewrite strategy that removes all spiders from a scalar Clifford diagram.

Pf First apply Clifford-simp. Then the only spiders left are $\circ$ and $\overset{\pi}{\circ}$. For these:

$$\circ \to 2 \cdot \boxed{} \qquad \overset{\pi}{\circ} \to \emptyset$$

$\blacksquare$

Q: What's left?

A: the scalar factor

$\boxed{D_0} \to \lambda_1 \cdot \boxed{D_1} \to \lambda_2 \cdot \boxed{D_2} \to \dots \to \lambda_n \cdot \boxed{} = \lambda_n \in \mathbb{C}$

<u>Application 1</u> (Efficient) strong simulation of Clifford circuits.

<u>Problem</u> For a circuit C, compute:

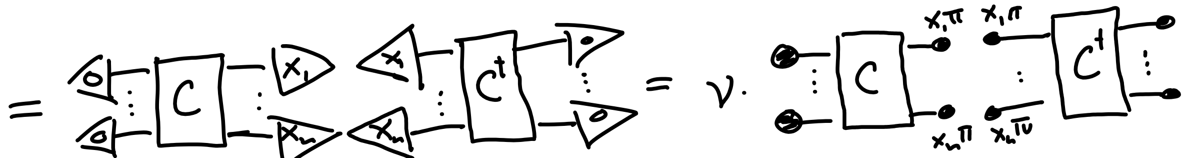
$$(*) \quad \mathrm{Prob}(x_1 \dots x_n \mid |\psi\rangle) \quad \text{where } |\psi\rangle = C|0\dots0\rangle.$$

... or more generally, for $k \leq n$, compute the marginal probability:

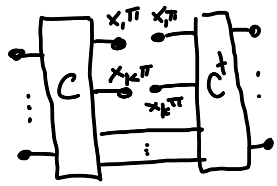
$$(**) \quad \mathrm{Prob}(x_1 \dots x_k \mid |\psi\rangle) = \sum_{x_{k+1}, \dots x_n} \mathrm{Prob}(x_1 \dots x_n \mid |\psi\rangle)$$

Born rule

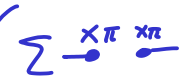
$$\mathrm{Prob}(x_1 \dots x_n \mid |\psi\rangle) := |\langle x_1 \dots x_n | C | 0 \dots 0 \rangle|^2$$



$$\Rightarrow \mathrm{Prob}(x_1 \dots x_k \mid |\psi\rangle) = \sum_{x_{k+1} \dots x_n} \mathrm{Prob}(x_1 \dots x_n \mid |\psi\rangle) = \nu' \cdot$$



ZX-diagram of (**)



$$\Big(\sum_x \ \ = 2 \cdot \sum_x |x\rangle\langle x| = 2I\Big)$$

Algorithm 1: For a circuit C:

1. Let D be the ZX-diagram of $\mathrm{Prob}(x_1 \ldots x_k \mid C|0\ldots 0\rangle)$.
2. Apply Clifford-simp to get a number.

Prop 1 Algorithm 1 terminates in polynomial time (in the # of qubits & gates of C).

Pf Assume basic diagram operations (add/remove spider/wire) take constant time. If C has n qubits & k gates, D has at most $S := 2 \cdot (2n + 2k) = 4(n+k)$ spiders. Then:

— Each rewrite removes 1 or 2 spiders, so there are at most $4(n+k)$ steps.

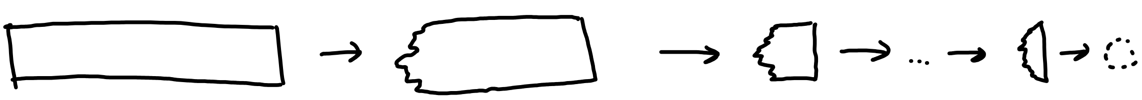
— Each step adds/removes at most $(4(n+k))^2$ edges, so Algorithm 1 performs $O((n+k)^3)$ basic graph operations. $\blacksquare$

Rem this is not optimal. A good choice of LC' and P' steps actually takes $O(n^2 k)$ time.
$\Rightarrow$ if $k \gg n$, this makes a big difference!

IDEA:

1. Avoid big spiders: 

2. Apply LC' & P' from left-to-right:



$\Rightarrow$ each step involves at most $O(n)$ spiders (hence $O(n^2)$ wires)

Lecture 12

Def A graph-like ZX-diagram is in AP-form if all interior spiders:

- have phase $\in 0, \pi$
- are only connected to boundary spiders.

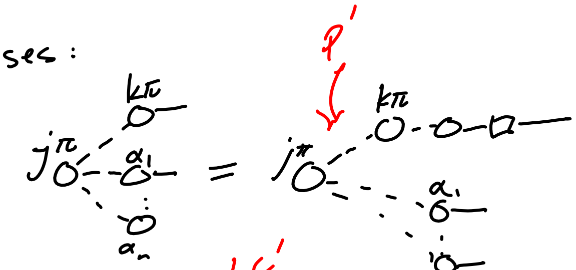
Def A ZX-diagram is in graph-state w/ local Clifford (GSLC) form if it has

* all Z spiders, fused as much as possible
* all spiders are connected to exactly 1 input (possibly via a 1-qubit Clifford unitary)

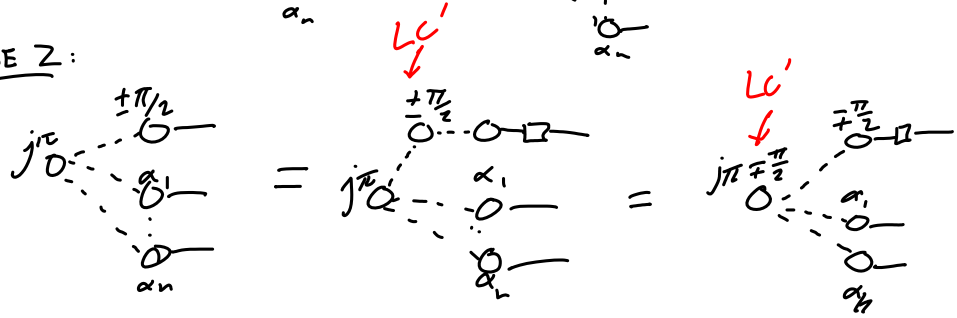
AP $\longrightarrow$ GSLC :

2 cases:

CASE 1:



CASE 2:



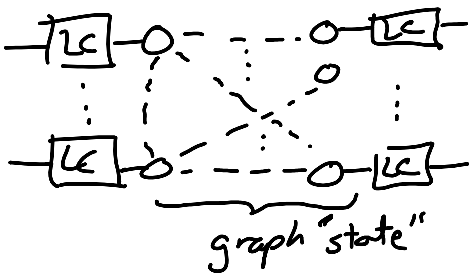
Application 2 Efficient synthesis of Clifford circuits.

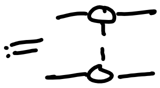
Clifford diagram $\rightarrow$ AP-form $\longrightarrow$ GSLC

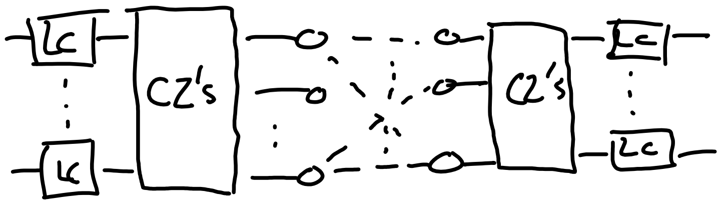
* only internal spiders are $\cancel{k\pi}$ * no internal spiders.

Algorithm 2 (Clifford re-synthesis)

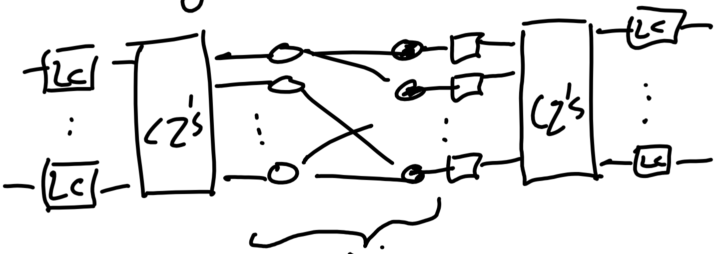
1. For an n-qubit Clifford circuit C, translate to ZX-diagram D.
2. Compute GSLC form.
3. Write as: $\leftarrow O(n)$



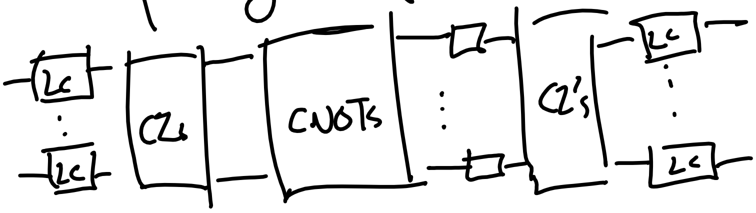
4. unfuse CZ gates  $\leftarrow O(n^2)$ and



5. colour-change:



4. extract parity map as CNOTs $\leftarrow O(n^2)$



Prop Any Clifford circuit can be written w/ at most $O(n^2)$ gates!

APPLICATION 3 Completeness of the ZX-calculus for Clifford diagrams.

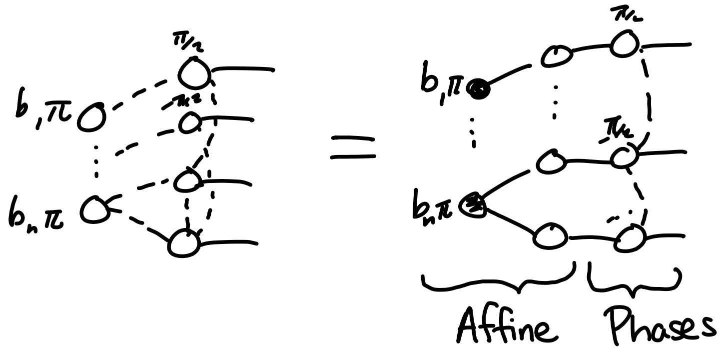
Thm (COMPLETENESS) For Clifford ZX-diagrams D_1, D_2, if $D_1 = D_2$ then $D_1 \overset{ZX}{=} D_2$.

matrices are equal

can (efficiently!) transform D_1 to D_2 with the ZX-calc.

IDEA: Look at the AP form.

Def A graph-like ZX-diagram is in AP-form if all interior spiders:
- have phase $\in 0, \pi$
- are only connected to boundary spiders.

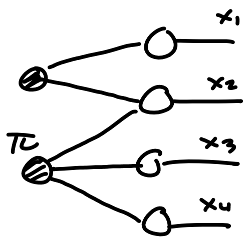


$$= \sum_{\vec{x},\, A\vec{x}=\vec{b}} e^{i\frac{\pi}{2}\cdot\phi} |\vec{x}\rangle$$

$A = \{\vec{x} \mid A\vec{x} = \vec{b}\}$ is an affine subspace of $\mathbb{F}_2^n$.

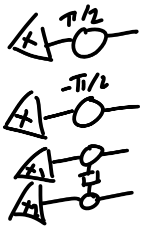
:= a solution to a set of linear eqns, e.g.:

$$A = \left\{ \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} \middle| \begin{array}{l} x_1 \oplus x_2 = 0 \\ x_2 \oplus x_3 \oplus x_4 = 1 \end{array} \right\} \iff$$



$$\approx \sum_{\vec{x},\, A\vec{x}=\vec{b}} |\vec{x}\rangle$$

ϕ is a phase polynomial

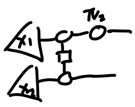


$$= e^{i\pi\cdot(\frac{1}{2}x)}|x\rangle$$

$$= e^{i\pi(-\frac{1}{2}x)}|x\rangle$$

$$= (-1)^{x_1 x_2}\cdot \; = e^{i\pi(x_1 x_2)}$$

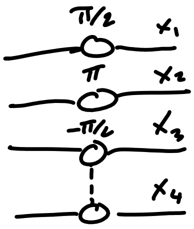
phase polynomial



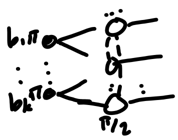
$$= e^{i\pi(x_1x_2)}\cdot \; = e^{i\pi(x_1x_2)}e^{i\pi(\frac{1}{2}x_1)} \; = e^{i\pi(x_1x_2+\frac{1}{2}x_1)}$$

$$U|\vec{x}\rangle = e^{i\pi\cdot\phi}|\vec{x}\rangle \text{ where } \phi = \tfrac{1}{2}x_1 - \tfrac{1}{2}x_3 + x_2 + x_3x_4$$

$U =$



Def An AP-form is in reduced AP-form if it is $\emptyset$ or A is in reduced echelon form and the polynomial ϕ only contains free variables from A.



$$= \sum_{\vec{x},\, A\vec{x}=\vec{b}} e^{i\pi\phi}|\vec{x}\rangle$$

free var x_3

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

echelon form

Prop Reduced AP-form is unique.

Pf (Linear algebra)

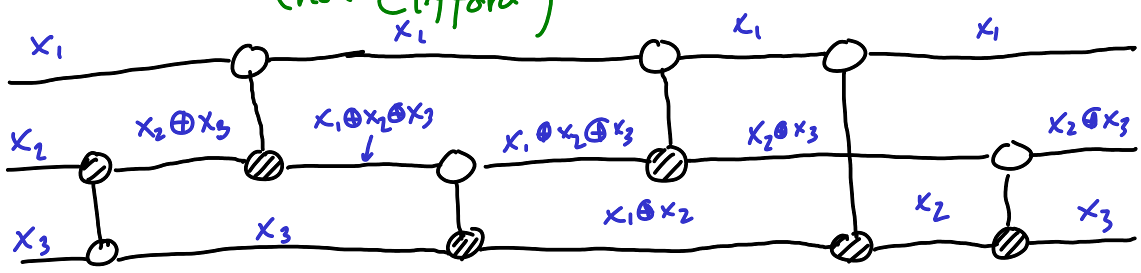
Prop For Clifford diag D, $D \overset{ZX}{=} D'$ ← reduced AP-form.

Pf (ZX can do Gaussian elimination!)

Cor Completeness!

CNOT + phase circuits

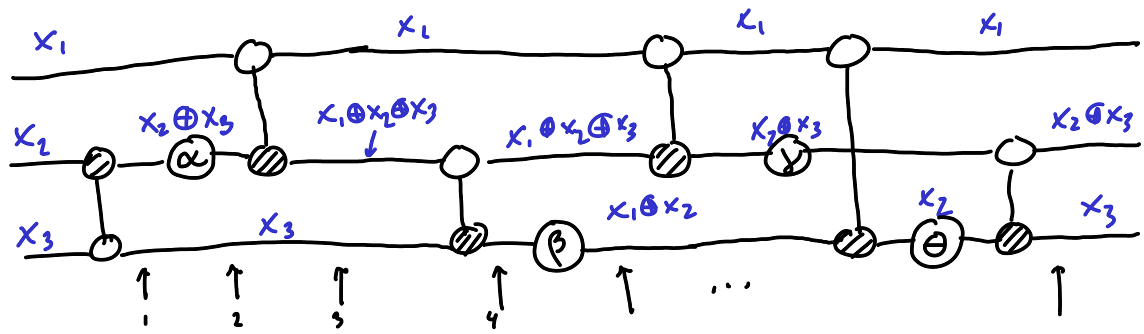
(non-Clifford)

$U =$ 

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

Q: What happens when we add phase gates?

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

$V =$ 

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

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

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

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

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

$$\mapsto \ldots$$

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

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.

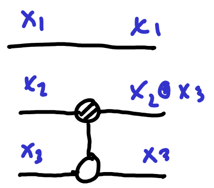
From the example above: $L = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$ and

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

phase-folding

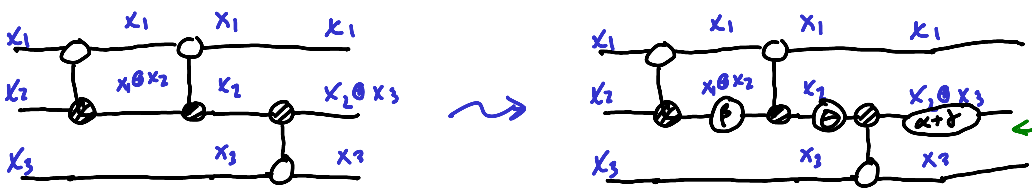
Q: can we re-synthesise a circuit for (L, ϕ)?

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-folding