Lecture 13

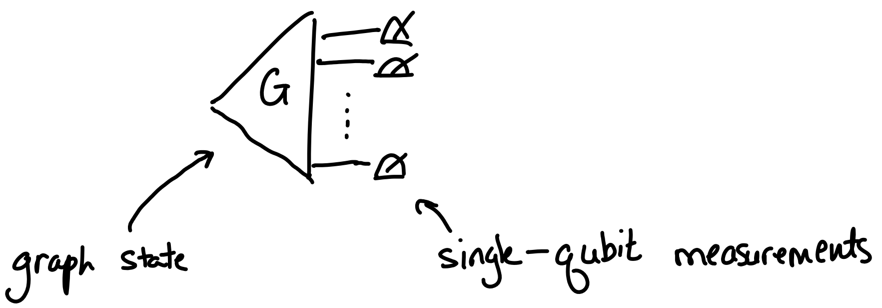
Measurement-based quantum computing (MBQC)

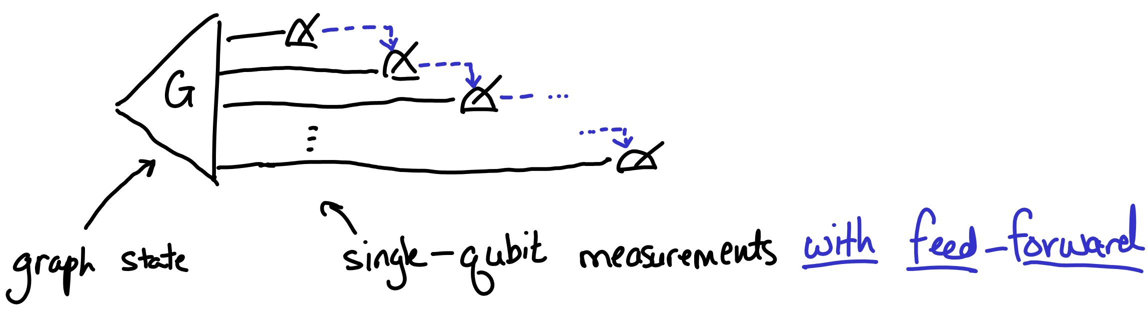
:= QC where measurements make up most of the computation.

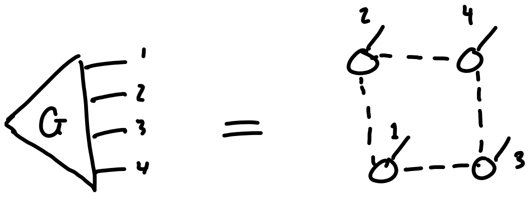
The "code" of MBQC is a measurement pattern:
:= measurement choices + classical control (feed-forward)

Several models:
- (gate teleportation)
- one-way model *
- hypergraph MBQC
- fault tolerant QC
  - lattice surgery (*)
  - topological FTQC
- ...

One-way model of MBQC (Raussendorf/Briegel 2001)

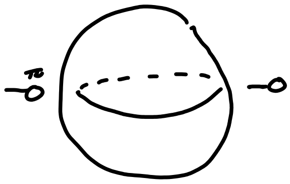




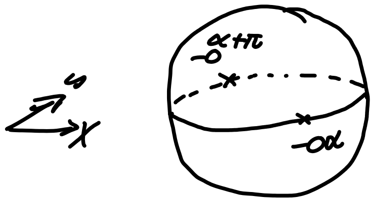


SINGLE-QUBIT MEASUREMENTS:

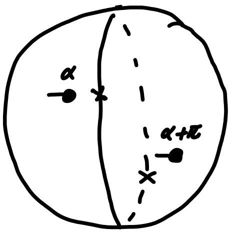
X - measurement: $\{ -\circ^{k\pi} \}_{k=0,1}$



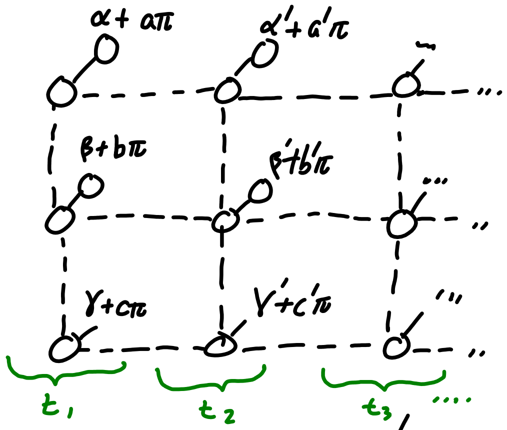
More generally: <u>XY-plane</u> measurements: $\{ -\circ^{\alpha+k\pi} \}_{k=0,1}$



Similarly, <u>YZ-plane</u> measurements: $\{ -\bullet^{\alpha+k\pi} \}_{k=0,1}$



(Z-measurements $\Rightarrow \alpha = 0$)

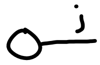
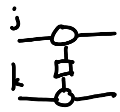
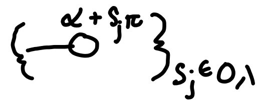
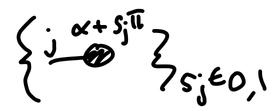
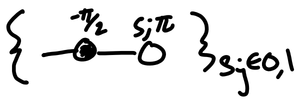
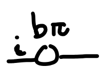


Feed-forward: $\alpha' = \alpha'(a,b,c)$ ← fn of (earlier) measurement outcomes (a.k.a. signals)

$\beta' = \beta'(a,b,c)$

......

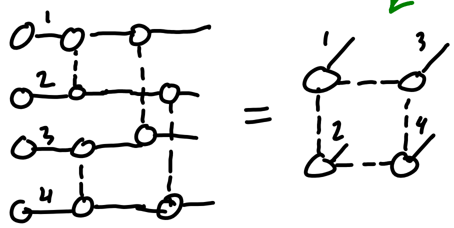
<u>Def</u> A <u>measurement pattern</u> for the one-way model consists of a sequence of instructions:

* $N_j :=$  prepare a <u>new</u> qubit in $|+\rangle$
* $E_{jk} :=$  entangle qubits j & k
* $M_j^{\alpha} :=$  measure qubit j in XY plane
  * store result in <u>signal</u> $s_j \in \{0,1\}$
  * $\alpha = \alpha(s_{k_1}, s_{k_2}, \ldots)$
* $M_j^{YZ,\alpha} :=$  " " " " YZ plane "
* $M_j^{XZ,\alpha} :=$  " " " - XZ plane "
* $Z_j^b :=$ , $X_j^b :=$  perform Pauli corrections, where
  * $b = b(s_{k_1}, s_{k_2}, \ldots)$

nodes edges

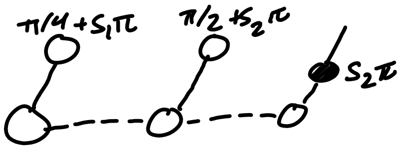
$$P := N_1; N_2; N_3; N_4; E_{12}; E_{34}; E_{13}; E_{24}$$

graph state



feed forward

$$Q := N_1; N_2; N_3; E_{12}; E_{23}; M_1^{\pi/4}; M_2^{\pi/2}; X_3^{s_2}$$



Def A measurement pattern is:

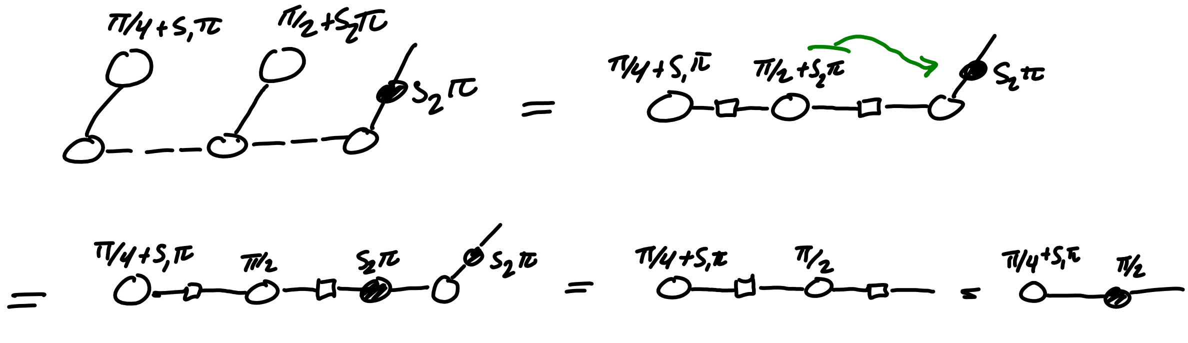
* runnable if all angles / corrections are fns of past measurement outcomes.

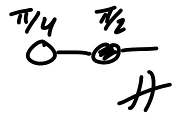
$$M_j^{\alpha}; \;\ldots\; ; Z_k^{s_j} \quad \text{OK}$$

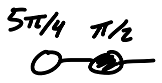
$$Z_k^{s_j}; \;\ldots\; ; M_j^{\alpha} \quad \text{BAD}$$

* deterministic if all choices of measurement outcomes give the same map (up to scalars)

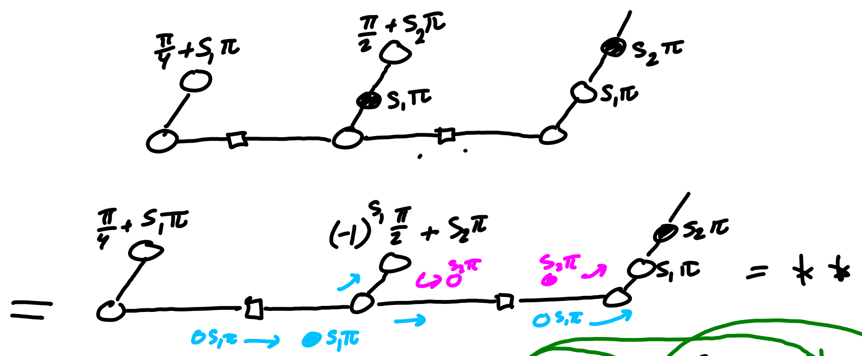
Q: runnable? ✓   deterministic ✗



$s_1 = 0 \Rightarrow * =$ 

$s_1 = 1 \Rightarrow * =$ 

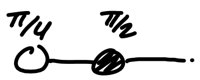
$$Q' := N_1; N_2; N_3; E_{12}; E_{23}; M_1^{\frac{\pi}{4}}; X_2^{s_1}; Z_3^{s_1}; X_3^{s_2}$$



$= **$

$$Q' = Q'' := N_1; N_2; N_3; E_{12}; E_{23}; M_1^{\frac{\pi}{4}}; M_2^{(-1)^{s_1}\frac{\pi}{2}}; Z_3^{s_1}; X_3^{s_2}$$

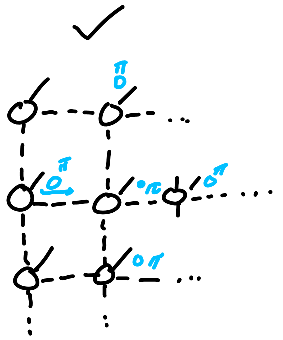
Q'' : runnable? ✓ deterministic? ✓

$s_1, s_2 \in 0,1 \Rightarrow ** =$ 

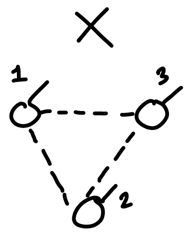
Lecture 14

<u>Question</u> Can I alway "push" errors forward in time?

<u>Answer</u>: It depends on the graph.



$t_1 < t_2 < t_3 \ldots.$



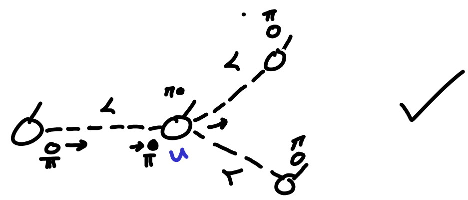
there is <u>no</u> time ordering for qubits $\{1,2,3\}$ that works.

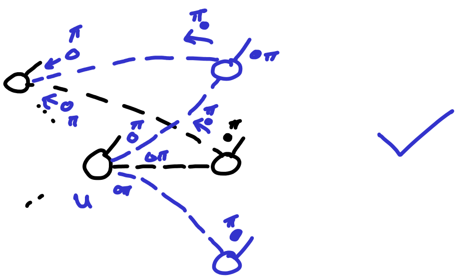
CLUSTER STATE
(:= graph state shaped like a square lattice)

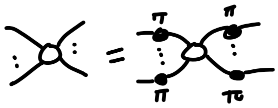
<u>Q</u>: how can we classify which graph states "work"?

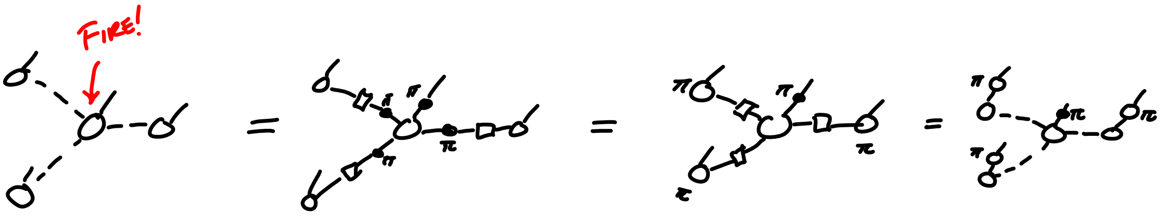
<u>IDEA</u>: 1. Fix a time-ordering $<$: $\begin{cases} \text{past}(u) := \{v \mid v < u\} \\ \text{future}(u) := \{v \mid u < v\} \end{cases}$

2. push errors from u into future(u) (without messing up past(u))





Equivalently, think about "firing" a spider with 

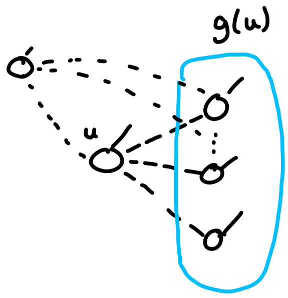


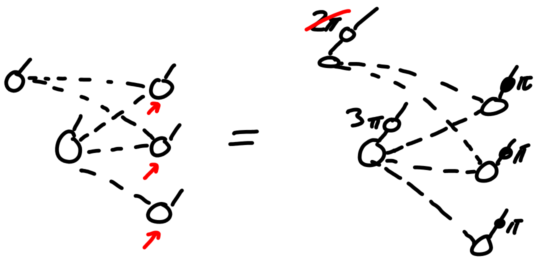
The game: for each u, find a set $g(u)$ that is:

(i) in the future of u

(ii) connected to u an <u>odd</u> number of times

(iii) Connected to the past of u an <u>even</u> number of times





Def An open graph is a graph G with a set of inputs $I_G \subseteq V_G$ and outputs $O_G \subseteq V_G$.

graph-like ZX-diag

open graph G

Def An open graph has generalised flow (gflow) if there exists a partial order $\preccurlyeq$ on V_G and a function $g : V_G \setminus O_G \to \mathcal{P}(V_G \setminus I_G)$ such that $\forall u$:

(i) $g(u) \subseteq \text{future}(u)$

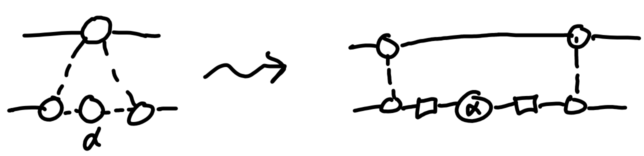
(ii) $g(u)$ connects to u an odd # of times

(iii) $\forall v \in V_G \setminus O_G$. if $v \neq u$, $v \notin \text{future}(u)$ then $g(u)$ connects to v an even # of times.

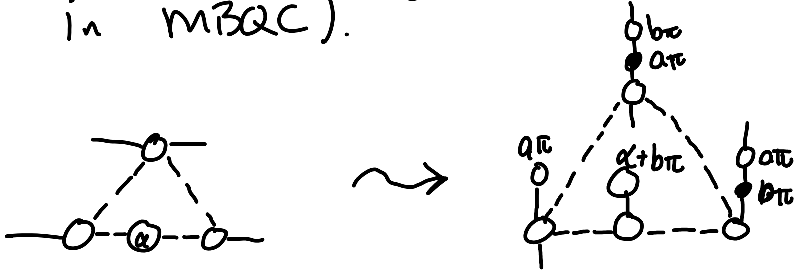
THM (Determinism) For any graph-like ZX-diagram D with gflow, there exists a runnable, deterministic pattern P that implements it.

$\Rightarrow$ There are at least 2 ways that a ZX-diagram can be "run" on a quantum computer:

1. If it can be transformed into a circuit.



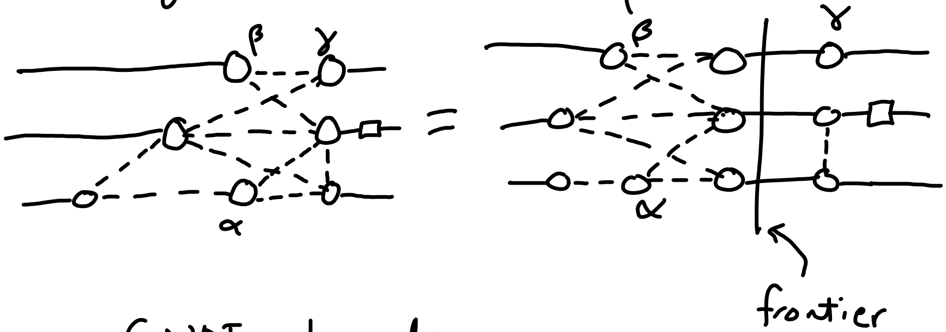
2. If it has gflow (hence can be implemented in MBQC).



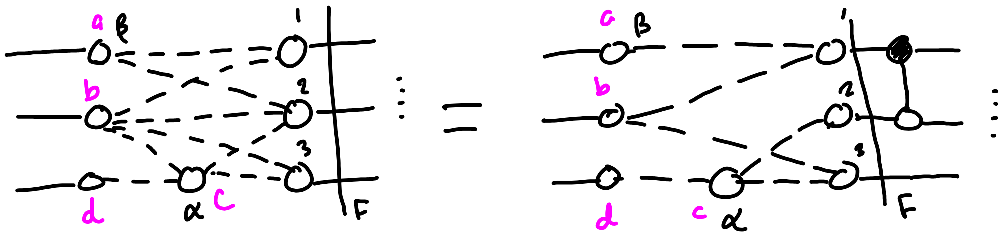
Now: 2. $\Rightarrow$ 1. (circuit extraction)

ALGORITHM (CIRCUIT EXTRACTION)

1. unfuse gates as much as possible:



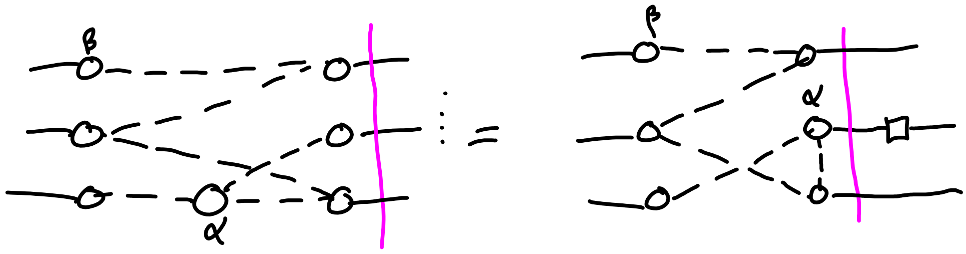
2. use CNOTs to do row operations until we get an "extractible" spider (= unit-vector row)

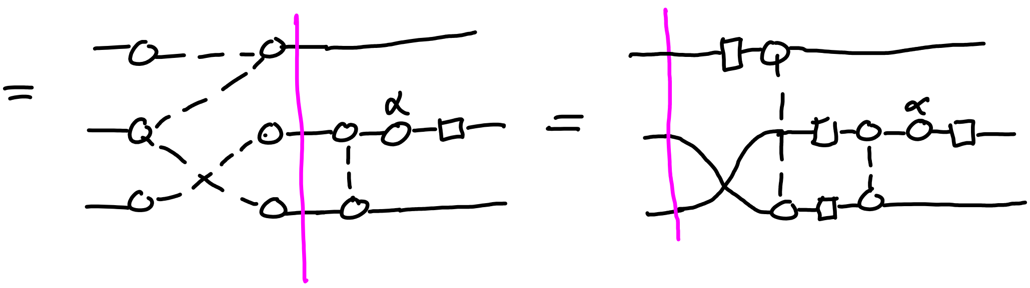


$$
\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array}
\begin{array}{c}
\begin{array}{cccc} a & b & c & d \end{array} \\
\begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}
\end{array}
\xrightarrow{R_2 = R_2 + R_1}
\begin{array}{c}
\begin{array}{cccc} a & b & c & d \end{array} \\
\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}
\end{array}
$$

extractible

3. Repeat 1 & 2 until nothing is left of the frontier.

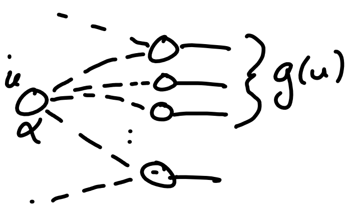




Thm If a ZX-diagram has gflow, CIRCUIT EXTRACTION terminates with a quantum circuit.

Pf Step 1 never adds spiders to the left of the Frontier, so s.t.s. Step 2 always removes a spider.

Take a maximal non-output u, w.r.t $<$.
Then $g(u) \subseteq \text{future}(u)$ must be all outputs:



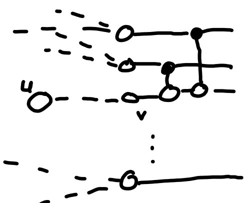
By gflow, the only node connected an odd # of times to $g(u)$ is u.

$$g(u)\left\{\begin{pmatrix} \boxed{E} & \boxed{E} & \boxed{O} & \boxed{E} & \cdots \end{pmatrix}\right.$$

If we add all the rows to a single row, then we get:

$$g(u)\left\{\begin{pmatrix} \boxed{0\,0\,1\,0\,0\cdots\cdots 0} \end{pmatrix}\right. \leftarrow v \in g(u)$$

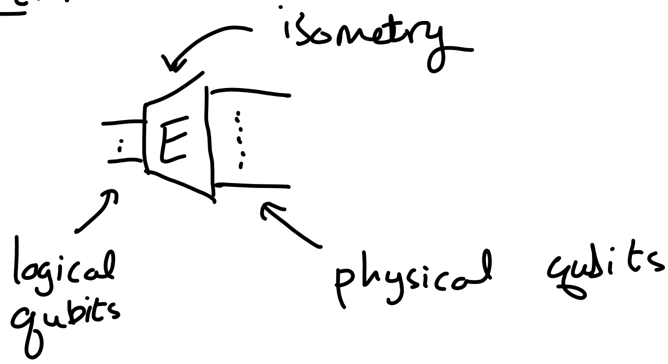
So, doing CNOTs ctrl'ed on a single $v \in g(u)$ to all other $v' \in g(u)$ gives:



← u becomes extractible

Extract & make u an output. The result <u>still</u> <u>has</u> <u>gflow</u> and there is one fewer spider left of the frontier. ∎

<u>Quantum</u> <u>error</u> <u>correction</u> works by encoding some <u>logical</u> <u>qubits</u> into a space of (more) <u>physical</u> <u>qubits</u>.



Q: Why?

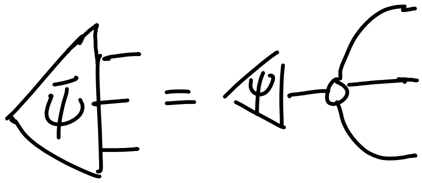
A: Because some errors can be <u>detected</u> and/or <u>corrected</u> using quantum measurements without destroying the logical state.

<u>Ex</u>. The GHZ code:

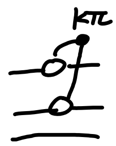
$$\underbrace{\mathbb{C}^2 = \text{span}\{|0\rangle, |1\rangle\}}_{2D} \xmapsto{E} \underbrace{\text{span}\{|000\rangle, |111\rangle\}}_{2D} \subseteq \underbrace{(\mathbb{C}^2)^{\otimes 3}}_{8D}$$

$$|\bar{0}\rangle := |000\rangle, \quad |\bar{1}\rangle := |111\rangle$$

MORE GENERALLY: $|\bar{\Psi}\rangle := E|\psi\rangle$.

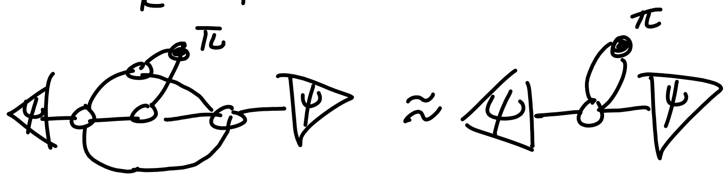


Suppose I measure ZZI:

$M_{ZZI} := \{ P_k :=$ 

$\}_{k=0,1}$

$$\text{Prob}(1 \,/\, |\bar{\Psi}\rangle) = \langle \bar{\Psi} | P_k | \bar{\Psi} \rangle$$

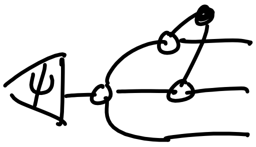
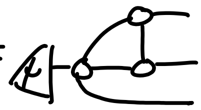
$=$ 

$\approx$ 

$= \emptyset$

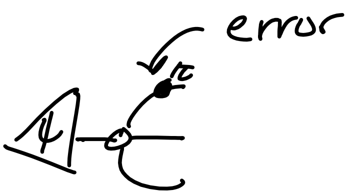
$$\Rightarrow \text{Prob}(0 \,|\, |\bar{\Psi}\rangle) = 1.$$

Also:

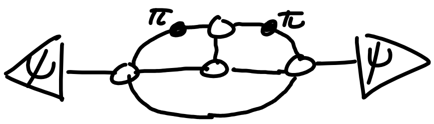
$P_0|\bar{\Psi}\rangle =$  $=$  $=$  $= |\bar{\Psi}\rangle$

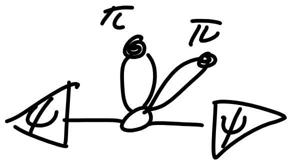
$\Rightarrow$ measuring ZZI does not disturb $|\bar{\Psi}\rangle$.

But



$\text{Prob}(0 \mid (X \otimes I \otimes I)|\bar{\Psi}\rangle) =$



$=$ 

$= \emptyset$.

$\Rightarrow \text{Prob}(1 \mid (X \otimes I \otimes I)|\bar{\Psi}\rangle) = 1$.

So a ZZI measurement can detect the error $X \otimes I \otimes I$.

Thm The GHZ code can detect (and correct) any error in the set $\{XII, IXI, IIX\}$.

bit-flip errors

Better codes correct more errors (e.g. "phase flips" like $ZII\cdots$, multi-qubit errors, etc.)