Tensor Network code : Part-I

Outline

- Tensor Network Notation (TNN)
- Tensor Operations
- Tensor Networks
- Stabilizer codes as tensors
- Generating new stabilizer codes
- Code distance via tensor network
- Maximum Likelihood decoding (MLD) via tensor network

Introduction to Tensor Network Notation (TNN)

Tensors?

- Generalisation of vectors and matrices

Dimension (d): Number of values an index can take

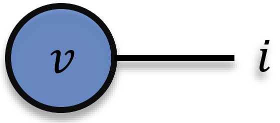
Rank (r) = Number of indices = Number of legs

Einstein Notation **TNN**

Vector (rank = 1)
An element of $\mathbb{C}^d$

$$v = \begin{bmatrix}1\\2\end{bmatrix}$$

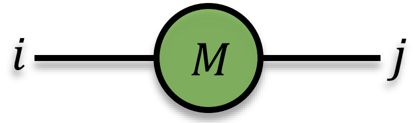
$$v_i$$



Matrix (rank = 2)
An element of $\mathbb{C}^{nxm}$

$$M = \begin{bmatrix}2 & 4\\1 & 3\end{bmatrix}$$

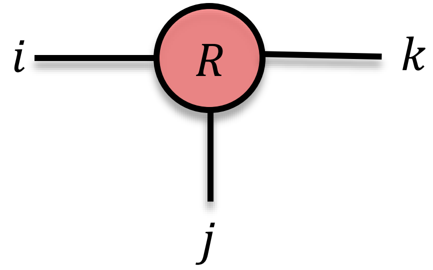
$$M_{ij}$$



Rank-3 tensor

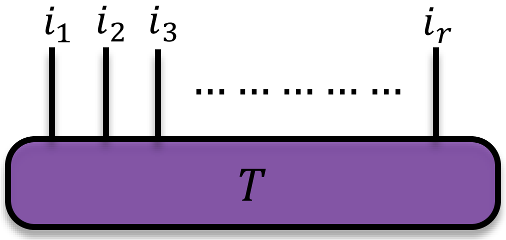
$$\mathrm{R} = \begin{bmatrix}2 & 4\\1 & 3\end{bmatrix}\begin{bmatrix}1 & 3\\5 & 2\end{bmatrix}$$

$$R_{ijk}$$



Rank-r tensor
An element of $\mathbb{C}^{d_1 \times d_2 \times \ldots \times d_r}$

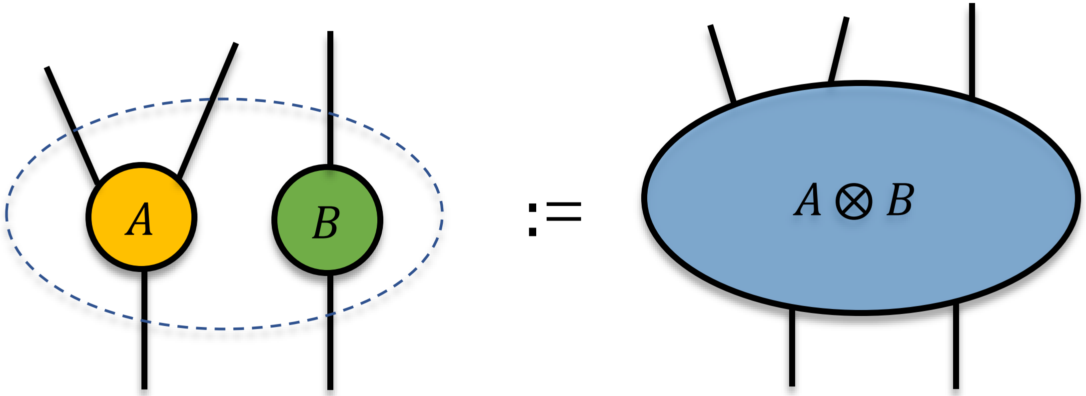
$$T_{i_1 i_2 i_3 \ldots i_r}$$



Tensors Operations

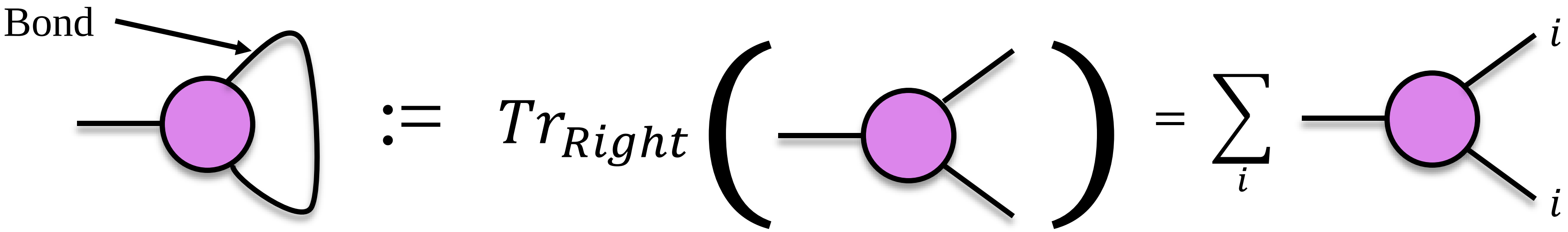
(1) Tensor product: Element-wise product of the values of each constituent tensor

$$[A \otimes B]_{i_1,\dots\, i_r, j_1 \dots\, j_s} := A_{i_1,\dots,i_r} \cdot B_{j_1,\dots,j_s}$$



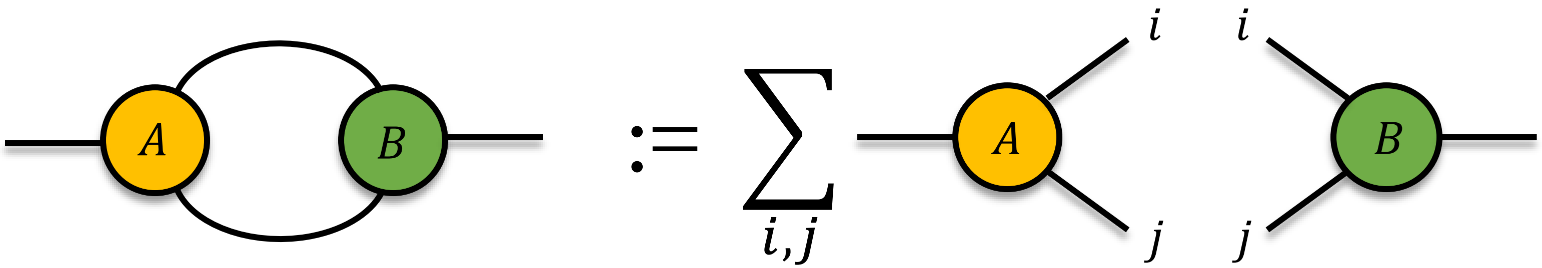
(2) Trace (partial): If x^{th} and y^{th} indices have identical dimensions ($d_x = d_y = bond\ dimension$) of tensor A; then,

$$[Tr_{x,y}\ A]_{i_1,\dots,i_{x-1},i_{x+1},\dots,i_{y-1},i_{y+1},\dots,i_r} := \sum_{\alpha=1}^{d_x} A_{i_1,\dots,i_{x-1},\alpha,i_{x+1},\dots,i_{y-1},\alpha,i_{y+1},\dots,i_r}$$



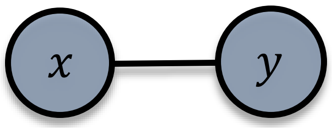
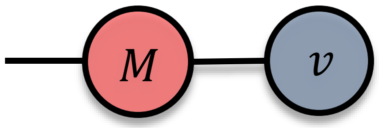
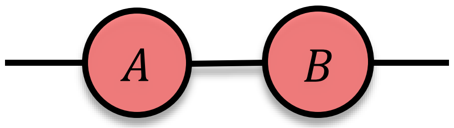
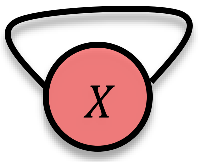
Tensors Operations

(3) Contraction: Tensor product followed by a trace between indices of two tensors.



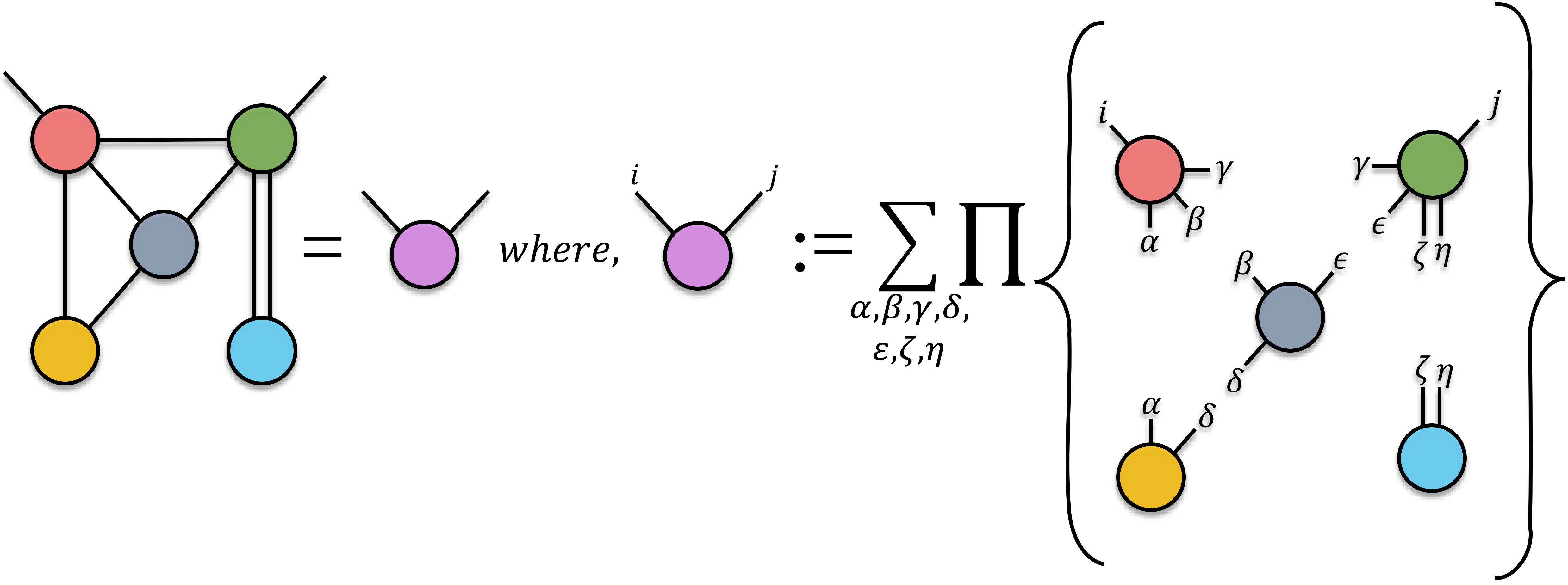
Internal Indices: *Bonds, here i* and *j*
External Indices: *free legs*

Examples:

Conventional	$\vec{x} \cdot \vec{y}$	$M\vec{v}$	AB	$Tr(X)$
Einstein	$x_\alpha y_\alpha$	$M_{\alpha\beta} v_\beta$	$A_{\alpha\beta} B_{\beta\gamma}$	$X_{\alpha\alpha}$
TNN				
	Inner Product	Matrix-vector multiplication	Matrix-matrix multiplication	Trace of a matrix

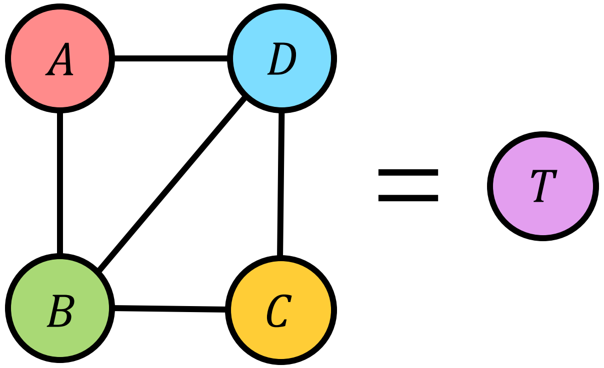
Tensor Networks

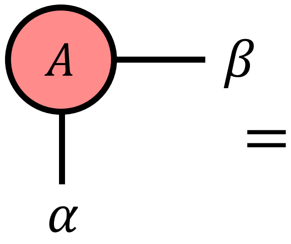
- A tensor network is a diagram which tells us how to combine several tensors into a single composite tensor.
- The **rank** of this overall tensor is given by the **number of unmatched/free legs.**
- The value for a given configuration of **external indices** (i.e. a fix value of i and j), is given by the product of values of the constituent tensors, summed over all **internal indices** ($\alpha, \beta, \gamma, \delta, \epsilon, \zeta, and\ \eta$).



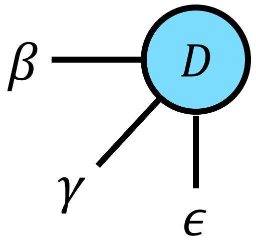
Tensor Networks

Example

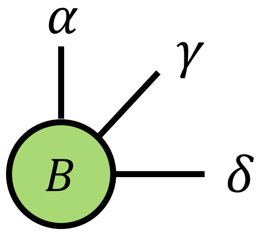




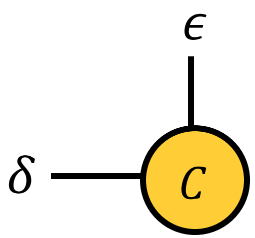
$= \beta^2 - 2\alpha$



$= -3^\alpha - 2\alpha$



$= -3^\alpha - 2\alpha$



$= \epsilon$

$$T = \sum_{\alpha,\beta,\gamma,\varepsilon,\delta=0}^{2} A_{\alpha,\beta} B_{\alpha,\gamma,\delta} C_{\delta,\epsilon} D_{\beta,\gamma,\epsilon} = \sum_{\alpha,\beta,\gamma,\varepsilon,\delta=0}^{2} (\beta^2 - 2\alpha)(-3^\alpha\gamma + \delta)\beta\gamma\epsilon^2$$

$$= 1080$$

Internal Indices: *All*
External Indices: *None*

All indices are 3-dimentional

Stabilizer codes as Tensors

- Represent Pauli operators by strings of integers.

$$I = \sigma^0, \quad X = \sigma^1, \quad Y = \sigma^2, \quad Z = \sigma^3$$

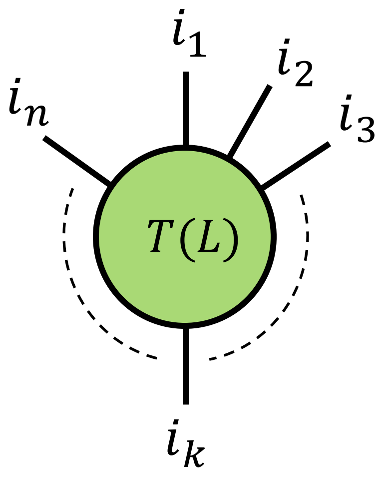
$$Ex. \quad XZZXI = \sigma^1\sigma^3\sigma^3\sigma^1\sigma^0 \textit{ as the string } (1,3,3,1,0)$$

- For a stabilizer code with **n physical qubits**, and **k logical qubits**, define the **rank-n tensor** for each logical operator L (ϵ Group of logical operators).

$$T(L)_{i_1,i_2,\ldots,i_n} = \begin{cases} 1 & if\ \sigma^{i_1}\otimes\cdots\otimes\sigma^{i_n} \in SL \\ 0 & otherwise \end{cases}$$

Where,

$i_k \in \{0,\ 1,\ 2,\ 3\}$
SL = Coset of S (Stabilizer group) with respect to logical L.
$L \,\epsilon$ Group of logical operators



Seed tensor

- T(L) is an indicator function for all operators in the class L.
- $T(\mathbb{1})_{i_1, i_2, \ldots, i_n}$ describes the stabilizer group.

Generating new stabilizer codes

- Contracting seed tensors generates larger codes.
- But how to check if the newly formed code is also a stabilizer code?

Theorem 1: Consider two code tensors $T(L)_{i_1,i_2,\ldots,i_n}$ and $T(L')_{h_1,h_2,\ldots,h_{n'}}$, which have n and n' physical qubits and k and k' logical qubits, respectively. We get new tensors describing a new stabilizer code by contracting indices (for simplicity, choose qubits 1 to l for both codes), **provided either one of these codes can distinguish any Pauli error on qubits 1 to $\boldsymbol{l}$.**

$$T_{new}(L \otimes L')_{(i_{l+1},\ldots,i_n,\, h_{l+1},\ldots,h_{n'})}$$

$$= \sum_{j_1,\ldots,j_l \in \{0,1,2,3\}} T(L)_{j_1,\ldots,j_l,i_{l+1},\ldots,i_n} T(L')_{j_1,\ldots,j_l,h_{l+1},\ldots,h_n}$$

- T_{new} describes a stabilizer code with $\boldsymbol{n+n'-2l}$ physical qubits and $\boldsymbol{k+k'}$ logical qubits.
- Stabilizer generators and logical operators for the new stabilizer code be found in $\mathrm{O}(n+n')$ time.
- *Theorem 1* allows us to iteratively build up very large codes with consistency guaranteed.

Generating new stabilizer codes

Julia package: https://github.com/qecsim/TensorNetworkCodes.jl

Example

```
# Start with the five-qubit code
tn_five_qubit = TensorNetworkCode(five_qubit_code())

# Contract with another five-qubit code (leg 5 of the first code with leg 1 of the second)
new_code = contract(tn_five_qubit, tn_five_qubit, [[5,1]]);

# Contract with another five-qubit code
new_code = contract(new_code, tn_five_qubit, [[8,1]]);

# Contract two existing legs
new_code = fusion(new_code,[1,11])

# Let's see what the code looks like
plot_code(new_code;use_coords=false)
```

```
new_code.stabilizers
```

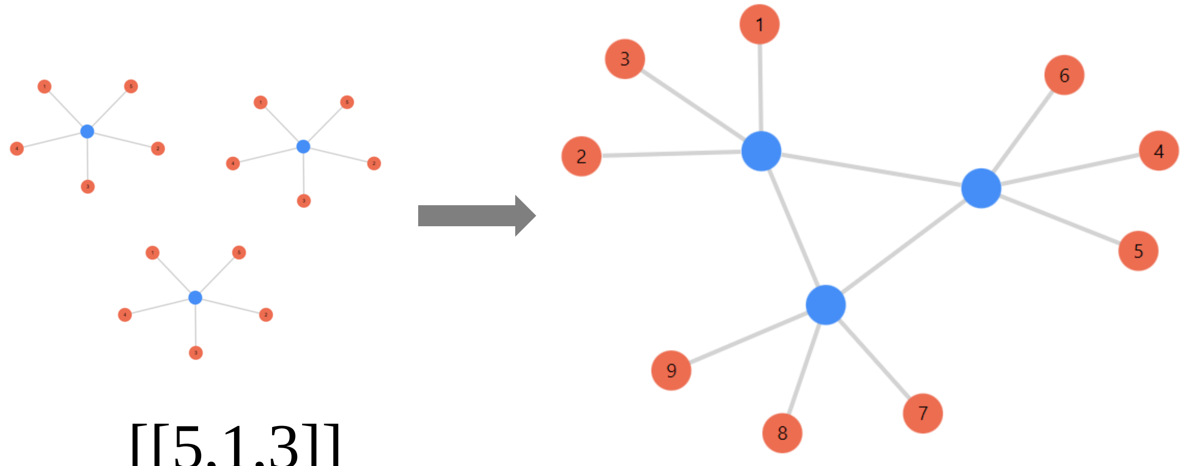
```
6-element Vector{Vector{Int64}}:
 [1, 3, 3, 3, 3, 1, 0, 0, 0]
 [2, 1, 1, 1, 1, 2, 0, 0, 0]
 [0, 0, 0, 1, 3, 3, 3, 3, 1]
 [3, 3, 1, 0, 0, 0, 1, 3, 3]
 [2, 2, 3, 1, 3, 3, 0, 1, 3]
 [3, 1, 0, 0, 1, 3, 1, 0, 1]
```

Distance calculation

```
dist = tn_distance(new_code)
n = num_qubits(new_code)
k = Int64(length(new_code.logicals)/2)

println("This is a [[$n,$k,$dist]] code!")
```

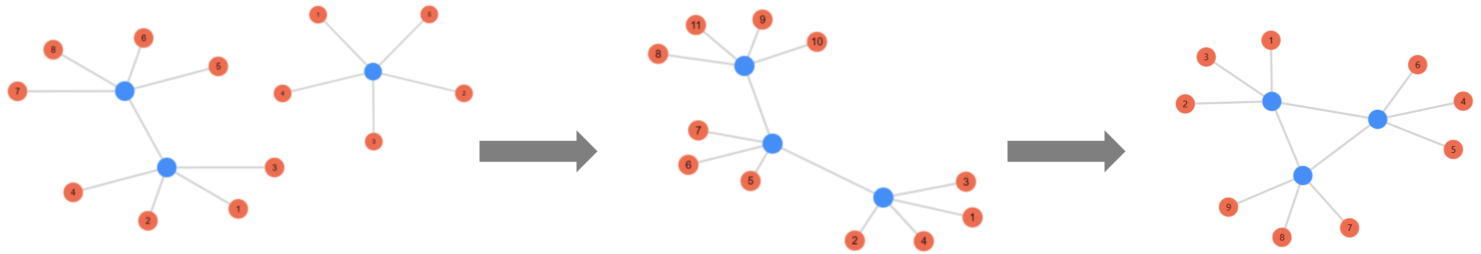
```
This is a [[9,3,3]] code!
```



[[5,1,3]]   [[9,3,3]]

Physical qubits: 3x5 − 2x3 = 9

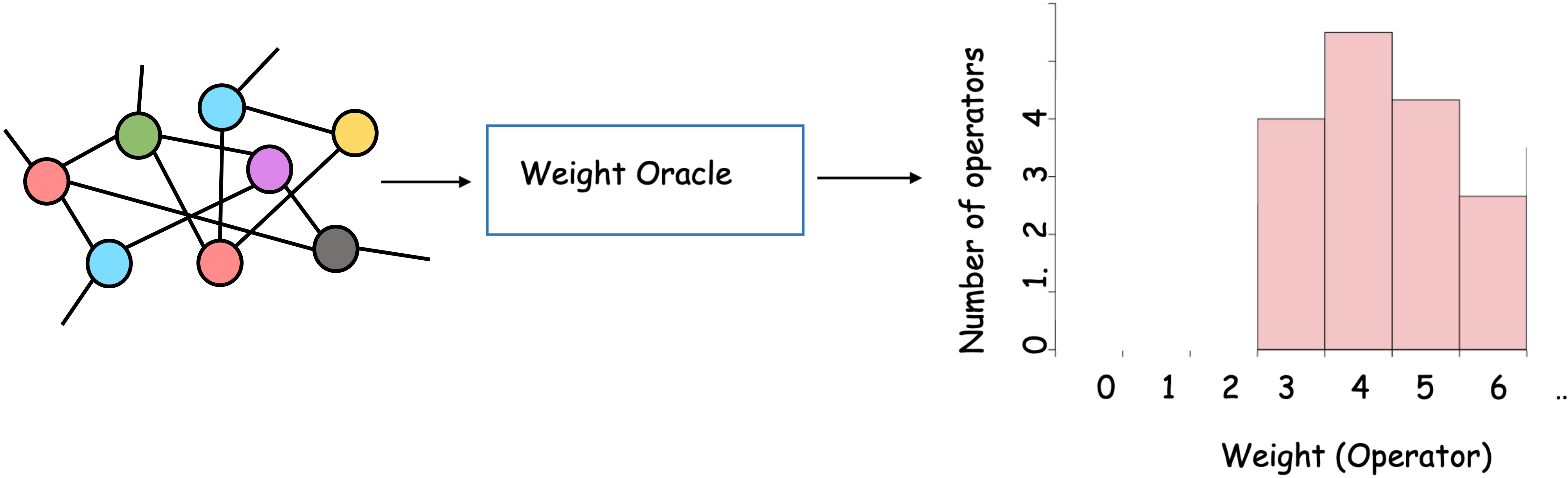
Logical qubits: 1+1+1 = 3



[[13,3]]   [[11,3]]   [[9,3]]

Iterations

'Code Distance' via Tensor Network



Requirement for an efficient Weight estimator :

- Existence of efficient contractable tensor structure in the input code.
- In general, finding such structure is NP-Complete but in certain cases we have efficient way to do this.

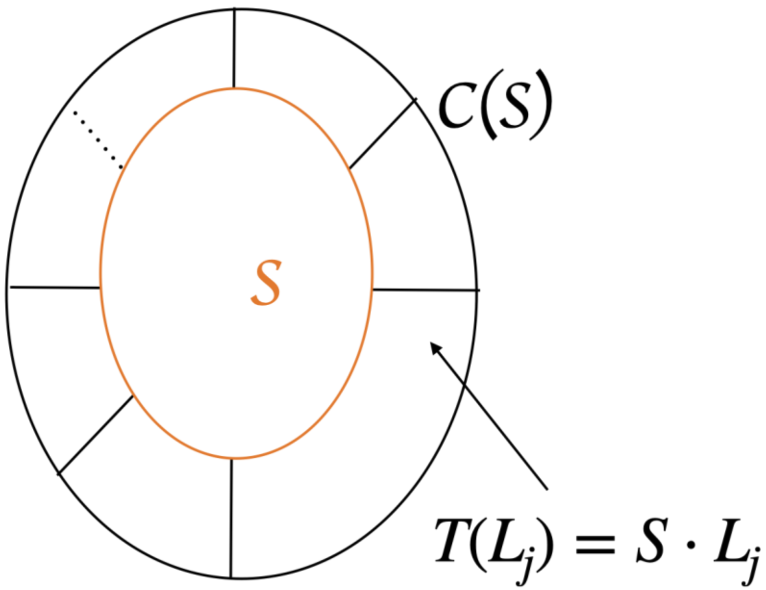
Construction of 'Weight-Oracle'

$$W_w^{i_1, \cdots, i_n} = \begin{cases} 1 \text{ if weight}(\sigma^{i_1} \otimes \ldots \otimes \sigma^{i_n}) = w \\ 0 \text{ otherwise,} \end{cases}$$

$$Ex: \quad W_3^{XXZZXX} = 0, \; W_3^{IXYZII} = 1$$

$$T(L)_{i_1, i_2, \ldots, i_n} = \begin{cases} 1 & if \; \sigma^{i_1} \otimes \cdots \otimes \sigma^{i_n} \in SL \\ 0 & otherwise \end{cases}$$

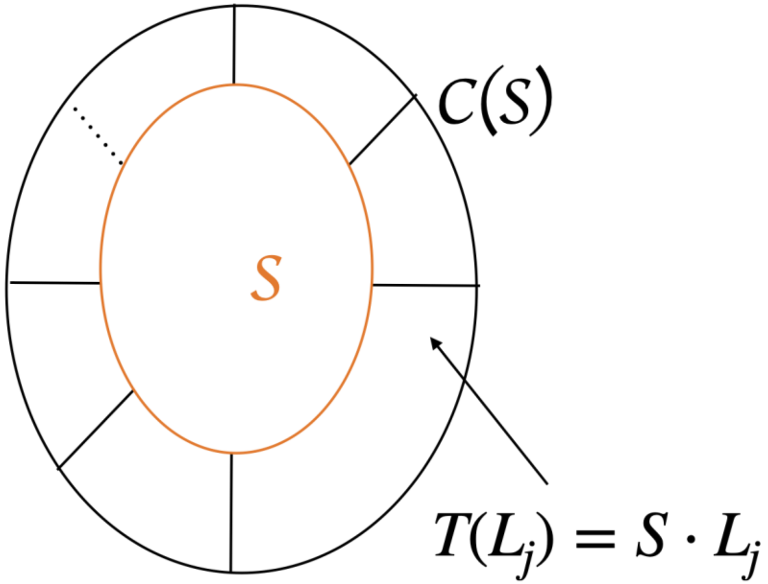
$$Ex: \quad T(XXX)_{XYZ} = 0, \; T(XXX)_{YYZ} = 1$$



(Continue....)

$$T(L)_{i_1,\ldots,i_n} = T^{l_1,\ldots,l_n}_{i_1,\ldots,i_n}$$ **[Equivalent Representation]**

$$C_w^{l_1,\ldots,l_n} = W_w^{i_1,\cdots,i_n} \, T^{l_1,\ldots,l_n}_{i_1,\ldots,i_n}$$



$C_w^{0,\ldots,0}$ = Number of stabilizer operator having 'w' weight.

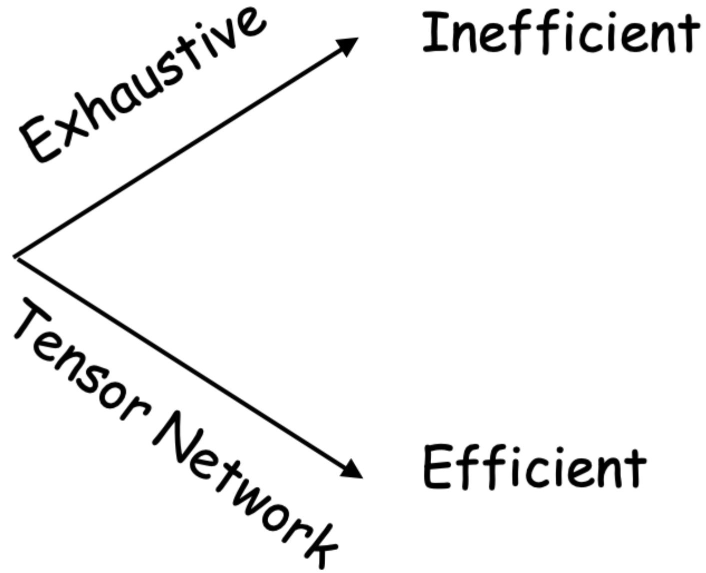
$\sum_{l_1,\ldots l_n} C_w^{l_1,\ldots,l_k}$ = Number of Centraliser operator having 'w' weight.

$\sum_{l_1,\ldots l_n} C_w^{l_1,\ldots,l_k} - C_w^{0,\ldots,0}$ = Number of C(S)/S Operators having 'w' weight.

$$\sum_{l_1,\dots,l_n} C_w^{l_1,\dots,l_k} \; - \; C_w^{0,\dots,0}$$

= Number of C(S)/S
Operators having 'w' weight.

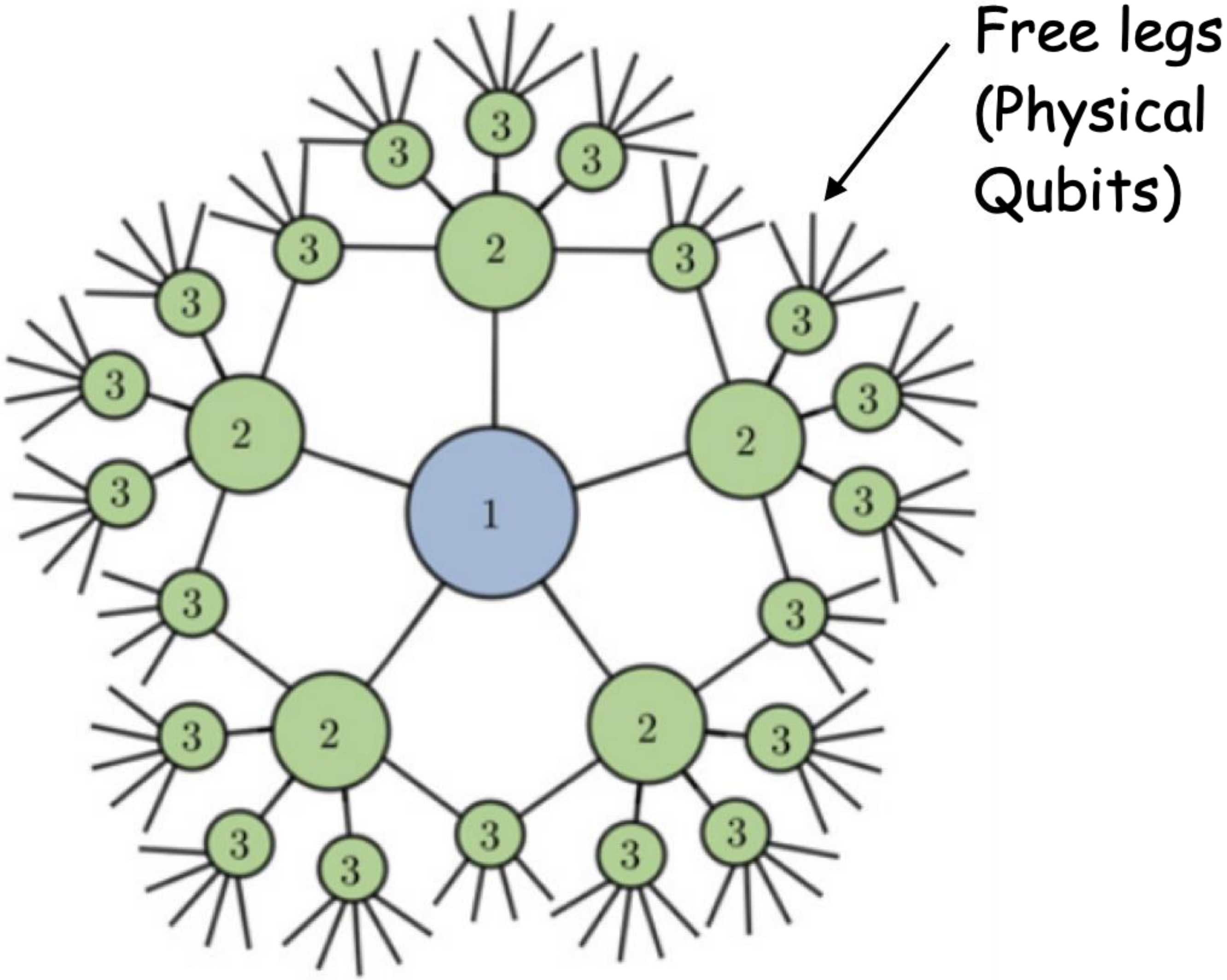
$$C_w^{l_1,\dots,l_n} \;=\; W_w^{i_1,\,\cdots\,,i_n}\; T_{i_1,\dots,i_n}^{l_1,\dots,l_n}$$

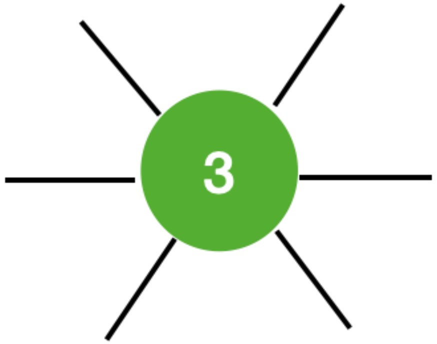


Reason for Efficiency of TN strategy:

- The above tensor can be computed efficiently if the original code tensor is contractable.
- We see an illustration of such efficient contraction for Holographic code

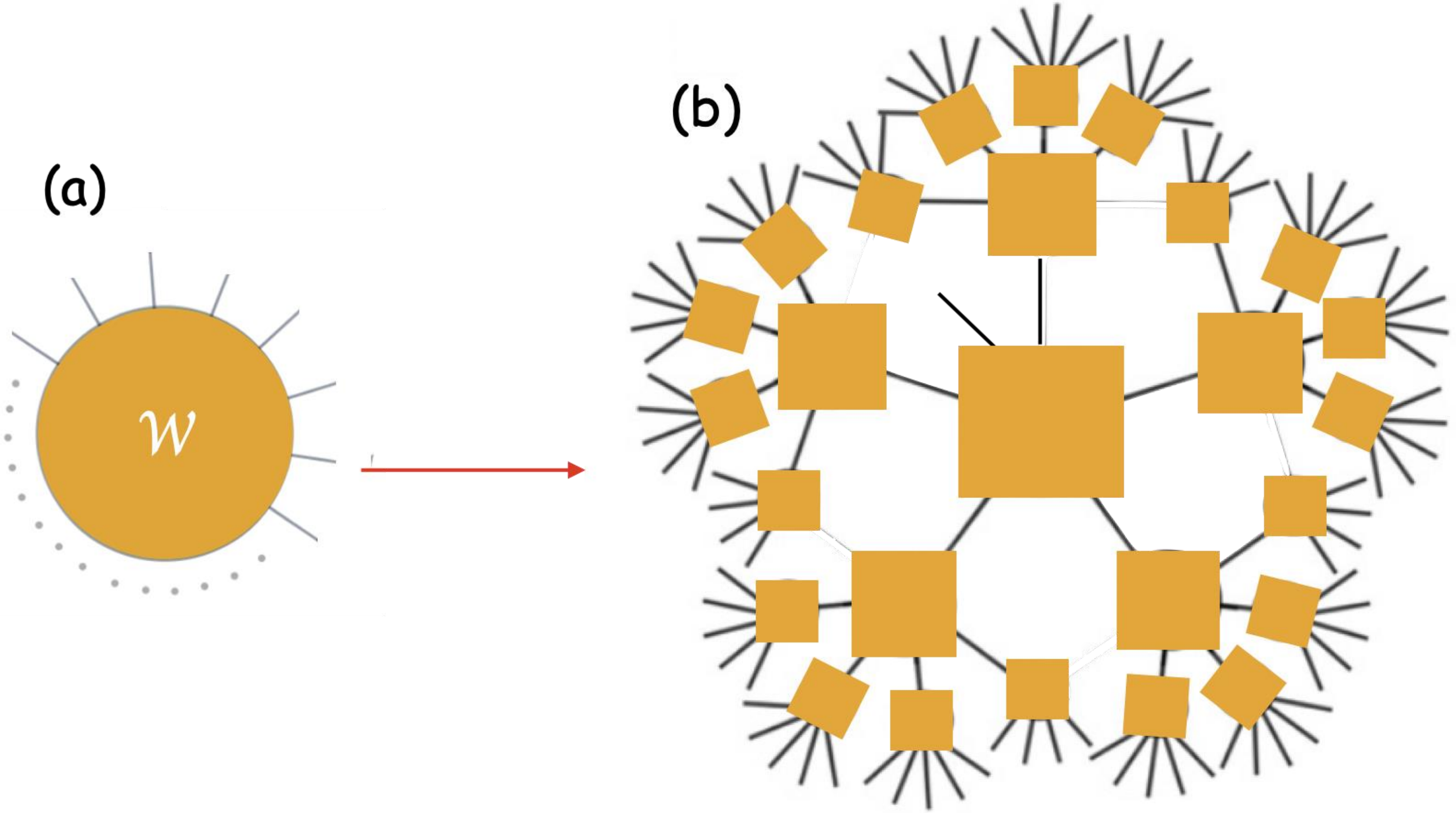
Example: Holographic code tensor [T(L)]





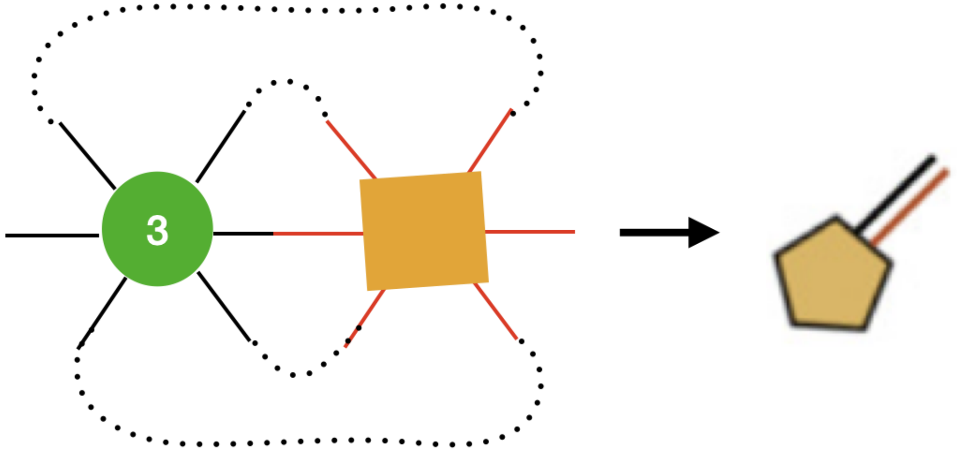
Seed code for the larger code

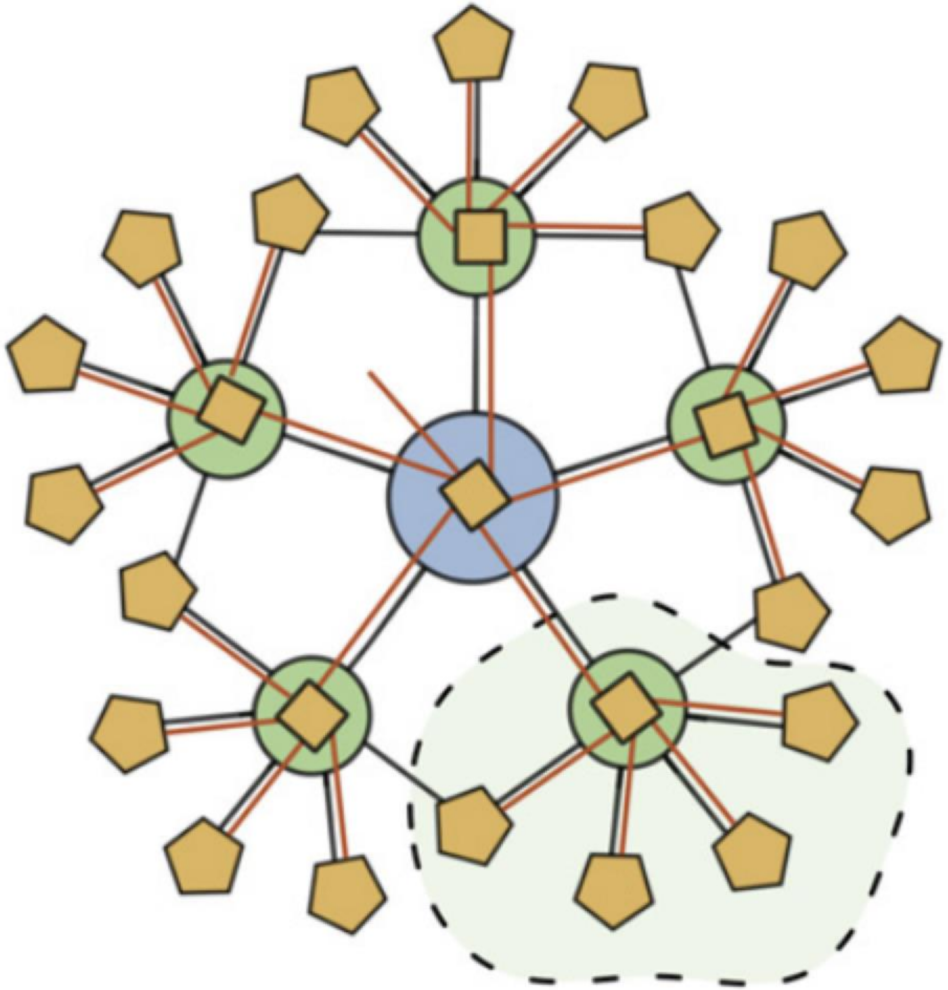
Chaining of Weight tensor (**W**) for efficient contraction.



- **Both right and left tensors are same tensor (number of legs are same.)**
- **The right one is given this form so that it can help in efficient contraction (mentioned in the next slide).**

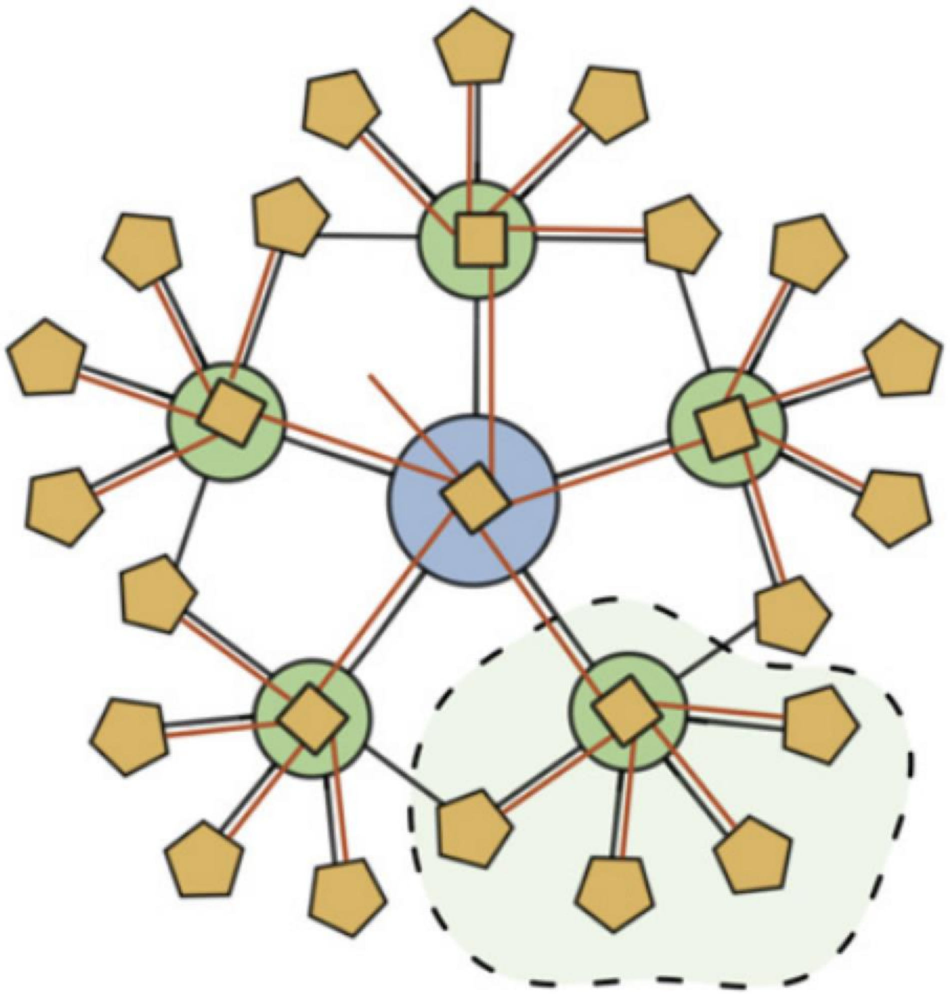
Valid contraction method





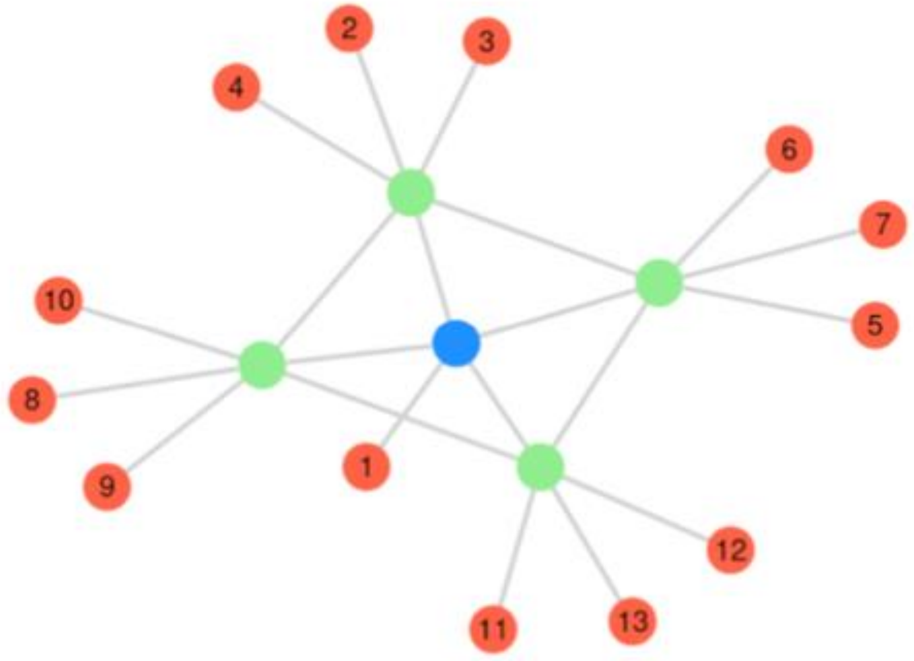
Contraction of W and T(L) tensors

Some observation on the contracted tensor



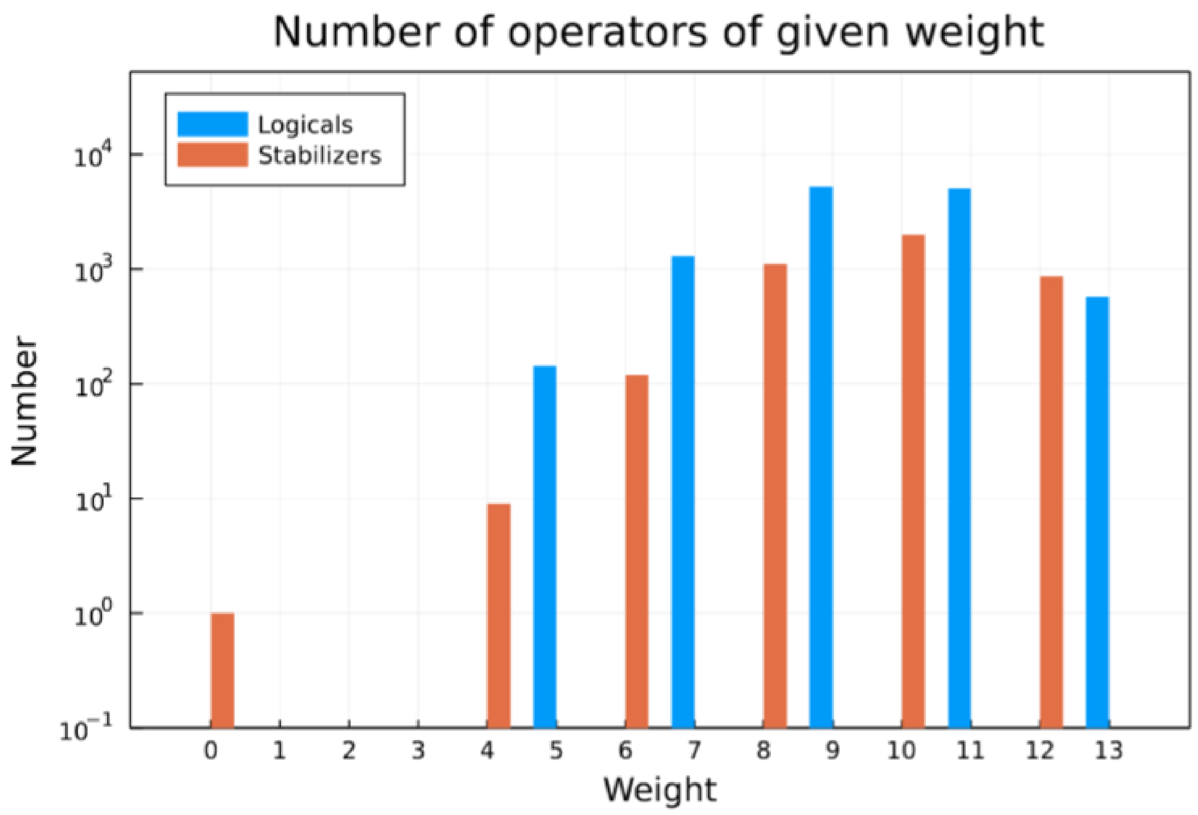
- One free leg (at the centre).
- It counts number of logical operators with weight 'w'.

Some statistics for [[13, 1, 5]] tensor code



[[13, 1, 5]]

Weight-Estimator
(TN method)



Useful for:

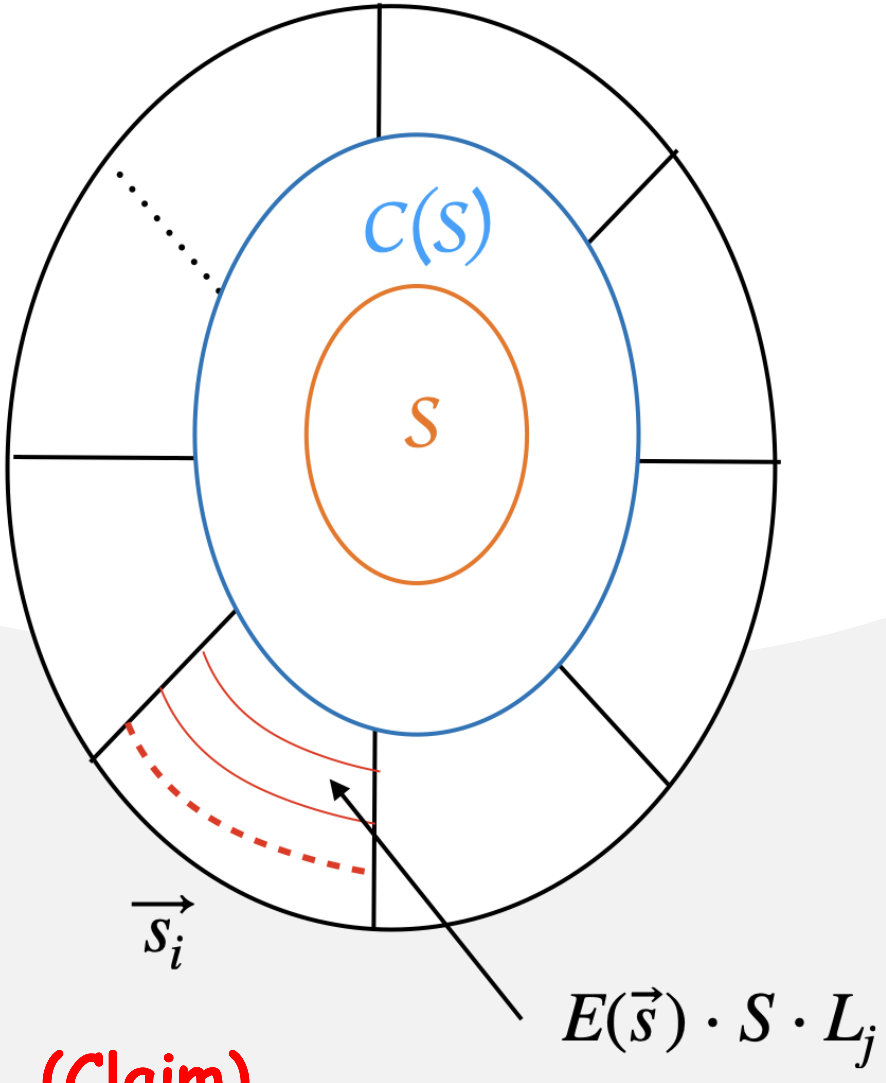
- MERA Tensor network
- Local "Log-Depth" Circuits

Maximum Likelihood Decoding (MLD): General Strategy

- Start with Syndrome Measurement:

$$\chi(L, \vec{s}) = \sum_{S} prob[E(\vec{s}) \cdot S \cdot L]$$

- **Goal:** $Arg\_Max_L \ \chi(L, \vec{s})$



- In general, MLD computation is computationally hard problem (NP-complete).
- (Recall) Toric code: MLD complexity $\sim \mathcal{O}(2^{L^2})$

$$\chi(L, \vec{s}) = \sum_{S} prob[E(\vec{s}) \cdot SL] = T(L)_{i_1,\ldots,i_n} \mathscr{E}(\vec{s})^{i_1, \cdots, i_n}$$

(Claim)

(Continue...)

$$\chi(L, \vec{s}) = \sum_{S} prob[E(\vec{s}) \cdot SL] = T(L)_{i_1,\ldots,i_n} \mathscr{E}(\vec{s})^{i_1, \cdots, i_n}$$

- We have just found tensor structure in MLD estimation.
- Efficient tensor contraction $\Longrightarrow$ Efficient MLD estimation.

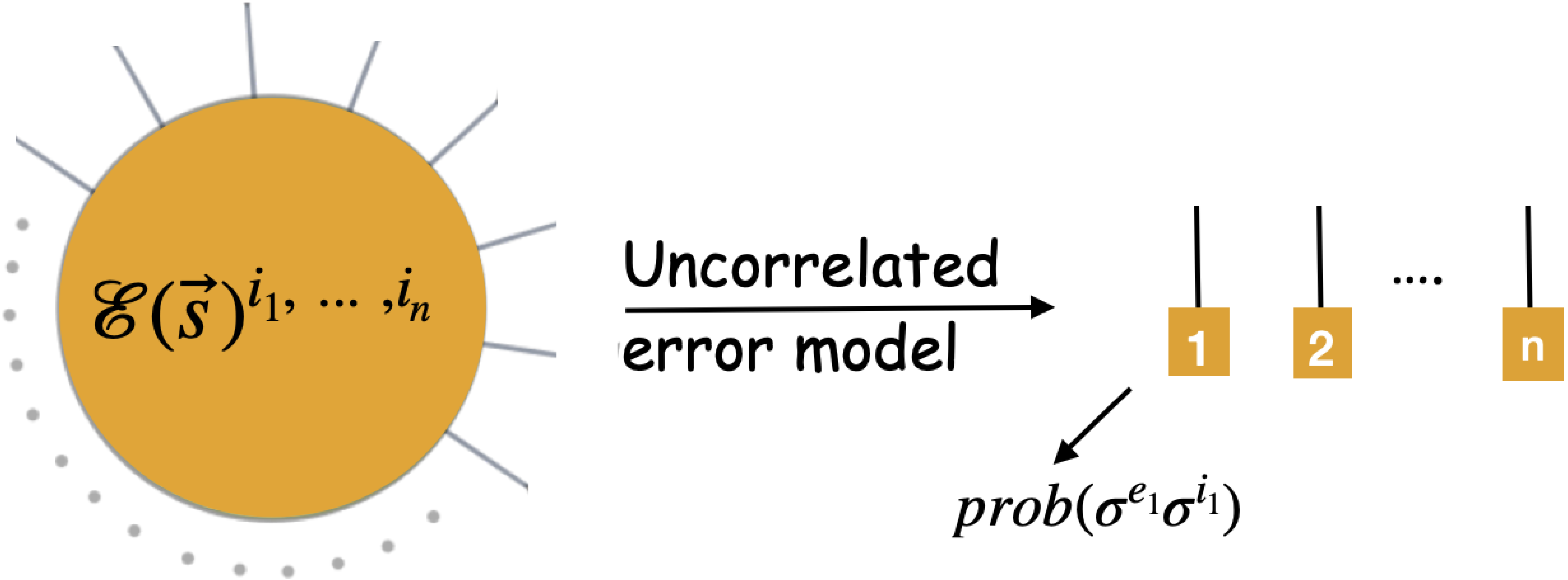
Simplification of the tensors under "Uncorrelated Quantum channel" assumption

$$\mathscr{E}(\vec{s})^{i_1, \cdots, i_n} =: prob(\sigma^{e_1}\sigma^{i_1} \otimes \ldots \otimes \sigma^{e_n}\sigma^{i_n})$$

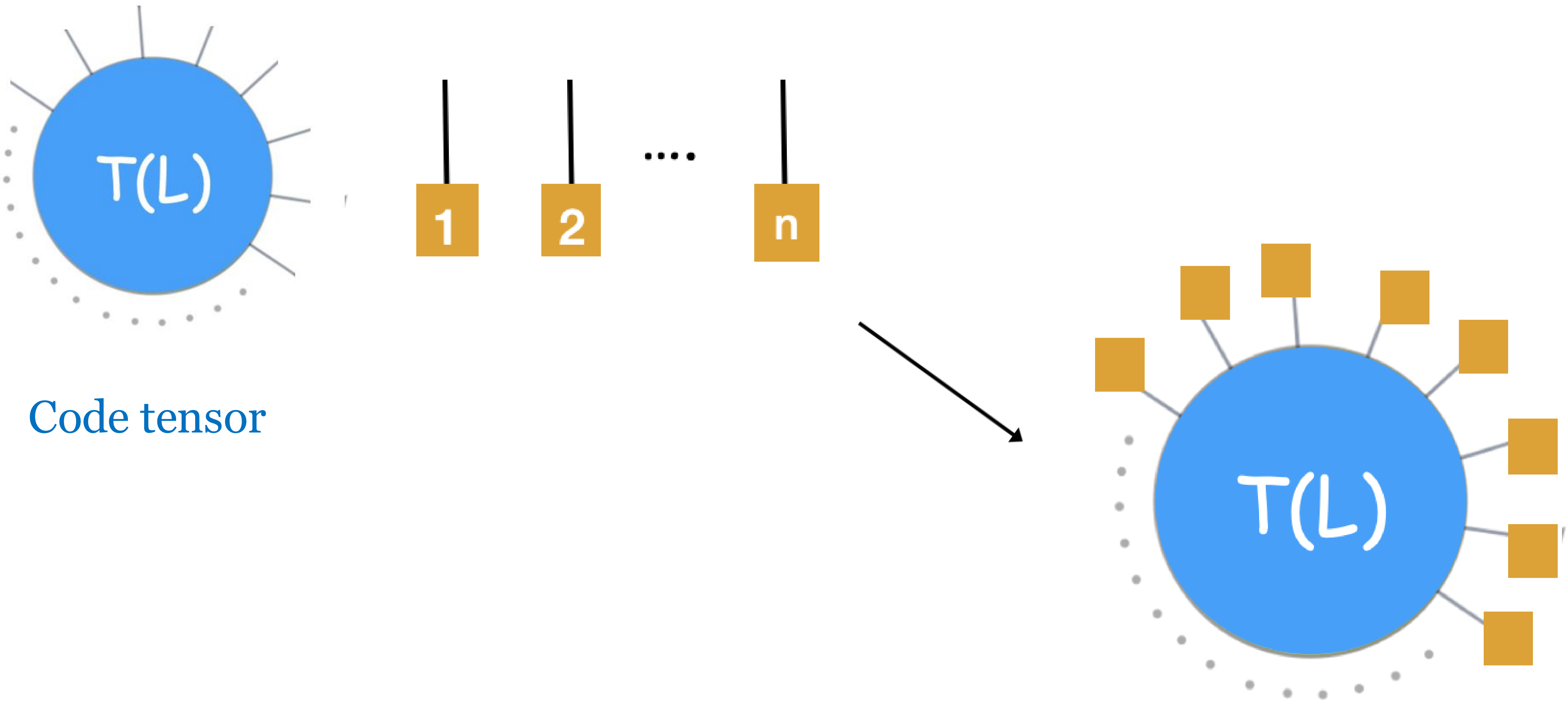
$$= prob(\sigma^{e_1}\sigma^{i_1}) \cdot prob(\sigma^{e_2}\sigma^{i_2}) \ldots prob(\sigma^{e_n}\sigma^{i_n})$$

Illustration in next page....

"Uncorrelated Quantum channel"



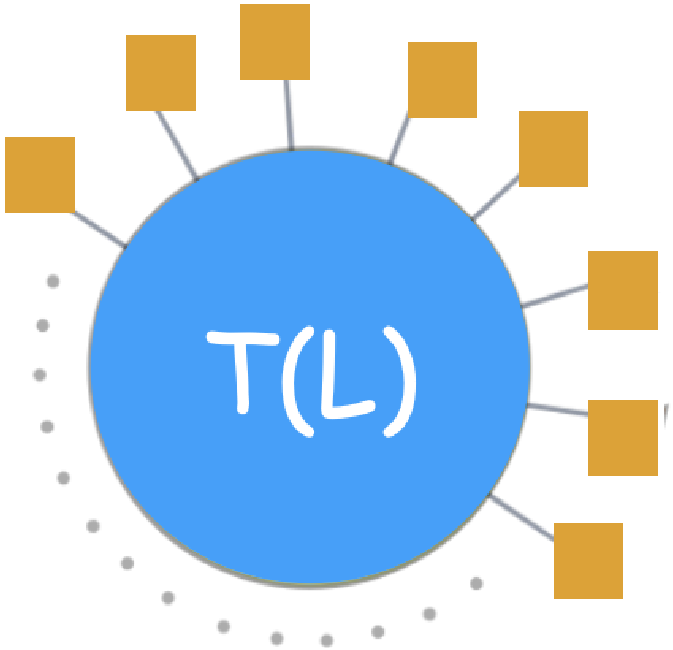
Efficient Contraction for uncorrelated error model



Code tensor

Result on contraction

Some observation on the contracted tensor



Result on contraction

- **No free legs (Scalar quantity)**
- **It represents** $\chi(L, \vec{s})$

Results

Codes	MLD (general) complexity	MLD (with TN) complexity	Assumptions
Planar Surface Codes	$\sim \mathcal{O}(\ 2^{n^2}\ )$	$\sim \mathcal{O}(\ n^2\ )$	Uncorrelated Quantum channel, and a few more.
Holographic Codes (Ex. HaPPY code, 3-Qutrit, etc)	$\sim \mathcal{O}(\ 2^{b \cdot n^2}\ )$	$\sim \mathcal{O}(n^{max[\frac{2.37}{c},\ 1]})$	Uncorrelated Quantum channel and a few more.

Remarks:

- Compare Toric code (a surface code) Edmond algorithms($\sim \mathcal{O}(\ E \cdot V^2\ )\ = \mathcal{O}(n^3)$)
- Some more context of uses: MERA, Concatenated and Convolutional codes.

References

- Farrelly, Terry, Tuckett, David K., and Thomas M. Stace. "Local tensor-network codes." *ArXiv*, (2021). Link
- Bridgeman, Jacob C., and Christopher T. Chubb. "Hand-waving and Interpretive Dance: An Introductory Course on Tensor Networks." *ArXiv*, (2016). Link
- Farrelly, Terry, Harris, Robert J., McMahon, Nathan A., and Thomas M. Stace. "Tensor-network codes." *ArXiv*, (2020). Link
- https://github.com/qecsim/TensorNetworkCodes.jl
- S. Bravyi, M. Suchara, and A. Vargo. Effcient algorithms for maximum likelihood decoding in the surface code. Phys. Rev. A, 90:032326, 2014. Link
- Ferris, Andrew J., and David Poulin. "Tensor Networks and Quantum Error Correction." *ArXiv*, (2013). Link
- https://errorcorrectionzoo.org/c/happy