LEARNING TO PLAY WITH QUANTUM LEGOS

Building Quantum Error Correcting Codes with Tensor Networks and Machine Learning

Vincent Su, UC Berkeley

Based on 2305.06378 w/ C. Cao, H.-Y. Hu, Y. Yanay, C. Tahan, Brian Swingle

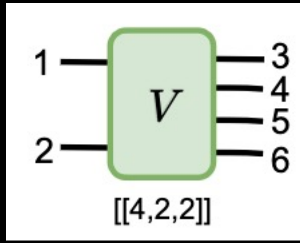
Quantum Information, Quantum Matter and Quantum Gravity, YITP 2023

OVERVIEW

- Quantum Legos
  - Quantum codes as tensors (Choi–Jamiołkowski)
  - Building a tensor network of codes
- Gamification
  - Turn code design into a game
  - Learn good moves by playing many times
- Results
  - Distance optimal CSS Code
  - State of the art protection against biased Pauli errors

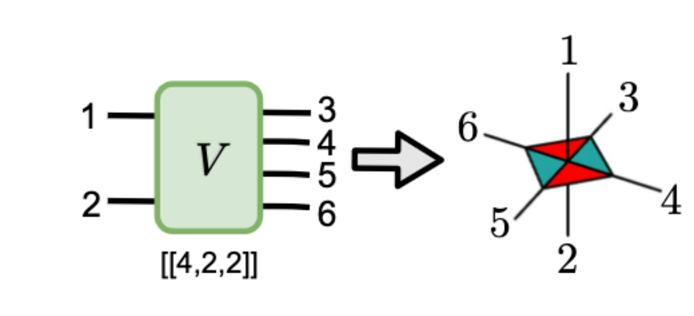
OUR FIRST LEGO (T6)

- Consider the following [[4,2,2]] code
- Stabilizers are $S = \langle X_3X_4X_5X_6, Z_3Z_4Z_5Z_6 \rangle$ $\quad s|\psi\rangle = |\psi\rangle$
- One logical operator is $X_1 = X_3X_4 = X_5X_6$



OUR FIRST LEGO (T6)

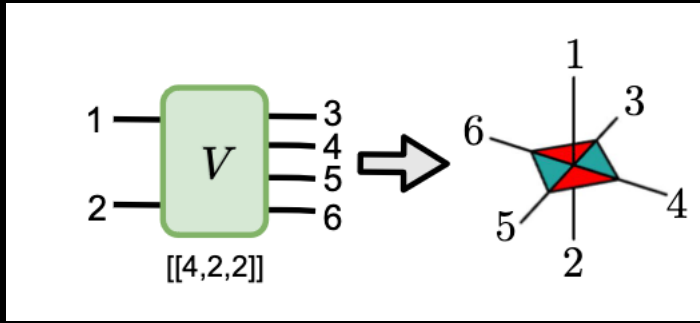
- Consider the following [[4,2,2]] code
- Stabilizers are $\langle X_3X_4X_5X_6, Z_3Z_4Z_5Z_6 \rangle$ $s|\psi\rangle = |\psi\rangle$
- One logical operator is $X_1 = X_3X_4 = X_5X_6$



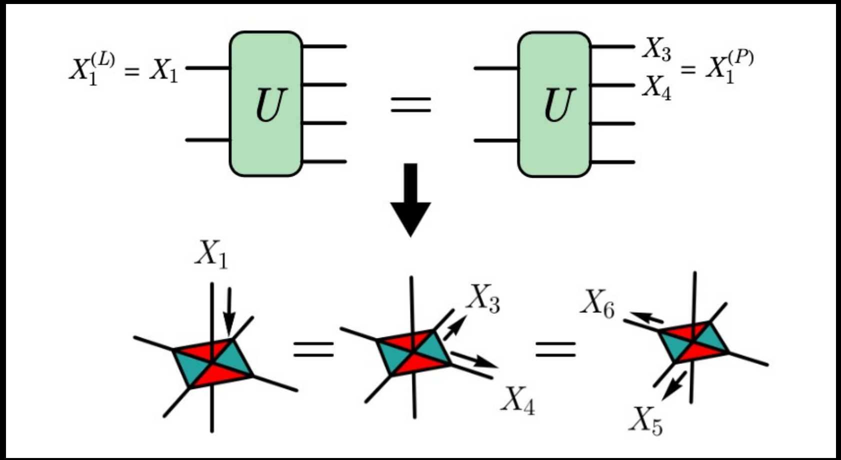
- Note that this isometry V can be interpreted as a stabilizer state!

OUR FIRST LEGO (T6)

- Consider the following [[4,2,2]] code
- Stabilizers are $\langle X_3X_4X_5X_6, Z_3Z_4Z_5Z_6 \rangle$ $s|\psi\rangle = |\psi\rangle$
- One logical operator is $X_1 = X_3X_4 = X_5X_6$

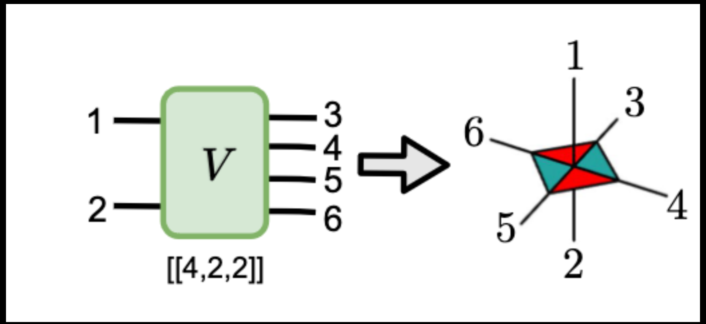


- Note that this isometry V can be interpreted as a stabilizer state!



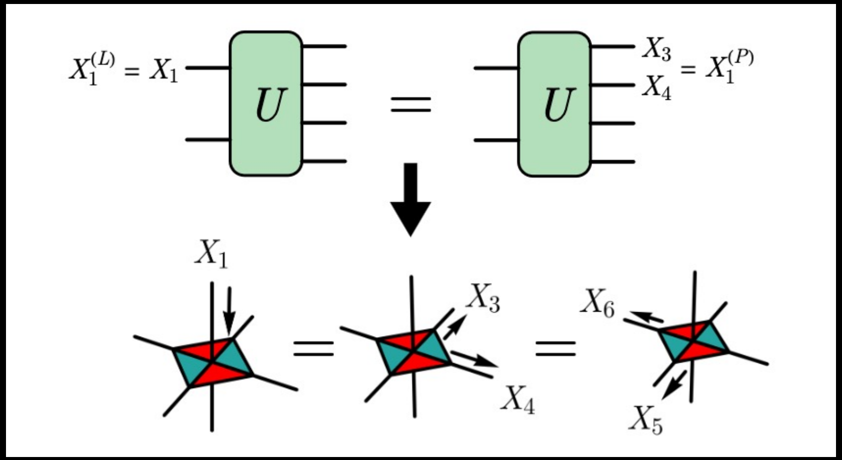
OUR FIRST LEGO (T6)

- Consider the following [[4,2,2]] code
- Stabilizers are $\langle X_3X_4X_5X_6, Z_3Z_4Z_5Z_6\rangle$ $s|\psi\rangle = |\psi\rangle$
- One logical operator is $X_1 = X_3X_4 = X_5X_6$



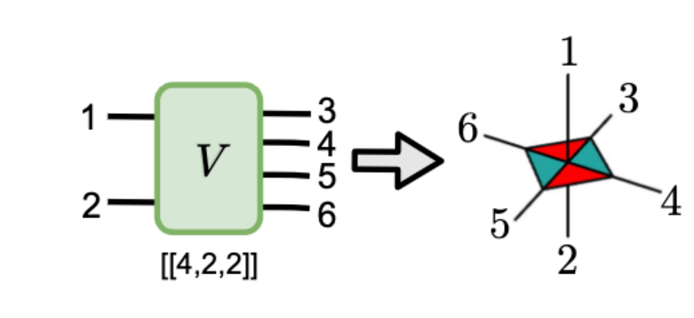
- Note that this isometry V can be interpreted as a stabilizer state!

- What are the stabilizers?
  - $I_1I_2X_3X_4X_5X_6$
  - $X_1X_3X_4$ $(= X_1^{(L)}X_1^{(P)} \sim I)$



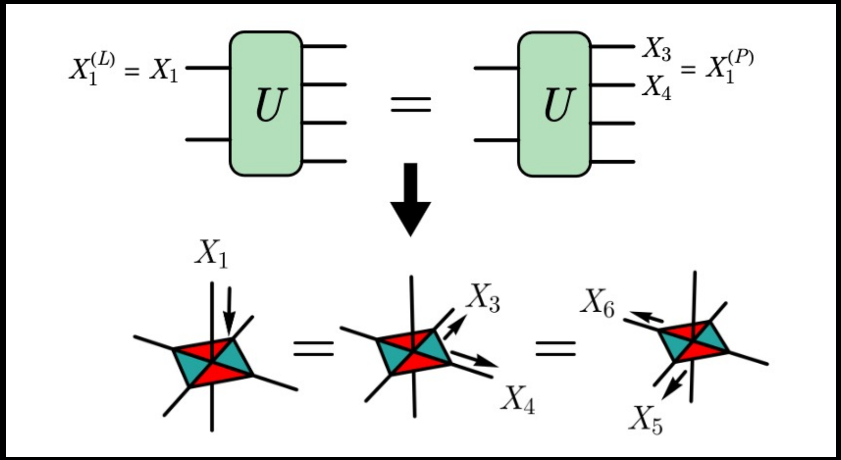
OUR FIRST LEGO (T6)

- Consider the following [[4,2,2]] code
- Stabilizers are $\langle X_3X_4X_5X_6, Z_3Z_4Z_5Z_6 \rangle$ $\qquad s|\psi\rangle = |\psi\rangle$
- One logical operator is $X_1 = X_3X_4 = X_5X_6$



- Note that this isometry V can be interpreted as a stabilizer state!

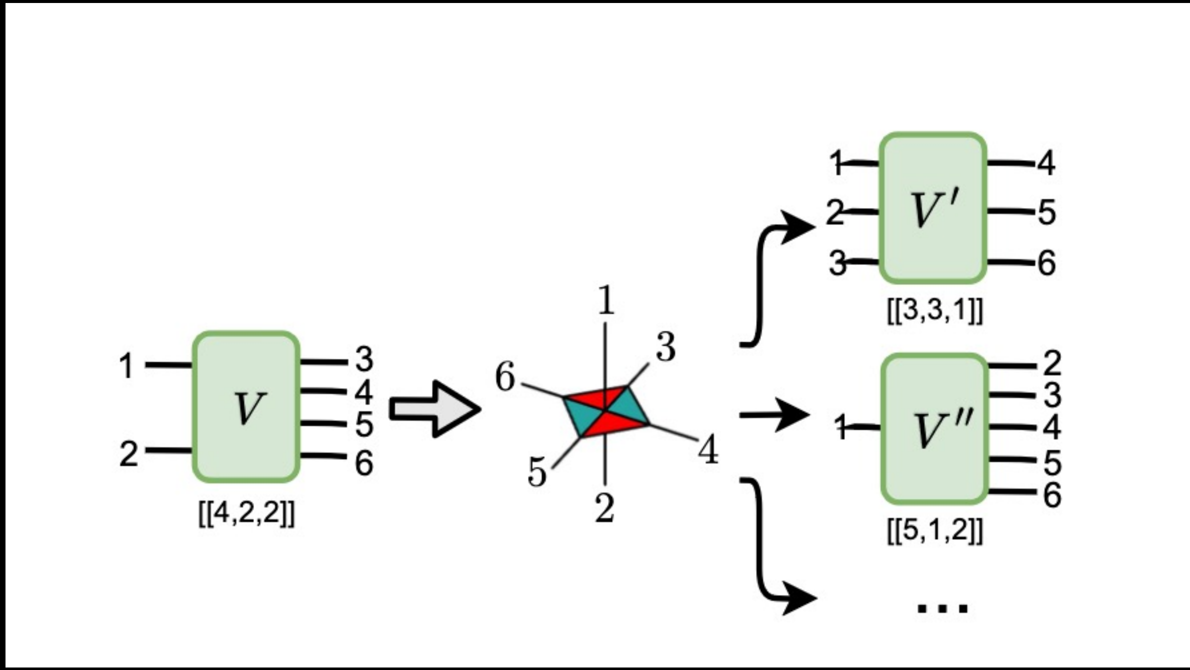
- What are the stabilizers?
  - $I_1I_2X_3X_4X_5X_6$
  - $X_1X_3X_4$ $(= X_1^{(L)}X_1^{(P)} \sim I)$



- No sense of directionality, all physical legs!

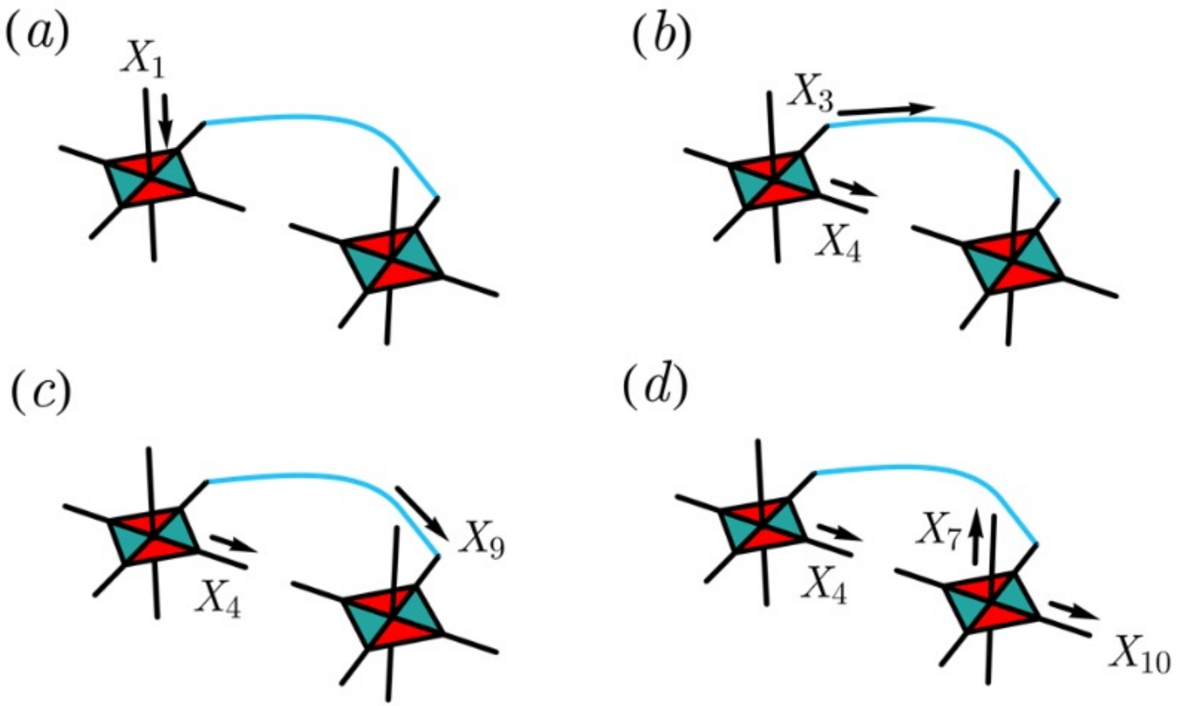
TENSOR → CODE(S)

- The same tensor can represent multiple codes!
- Ask me later for details



ASSEMBLY INSTRUCTIONS
[AGES 6+]

- Glue legs on two copies of the T6 lego
- Demand that stabilizers acting on those legs match
- Formally, amounts to a bell projection
- Alternatively, can do operator pushing
  - New stabilizer $X_1 X_4 X_7 X_{10}$



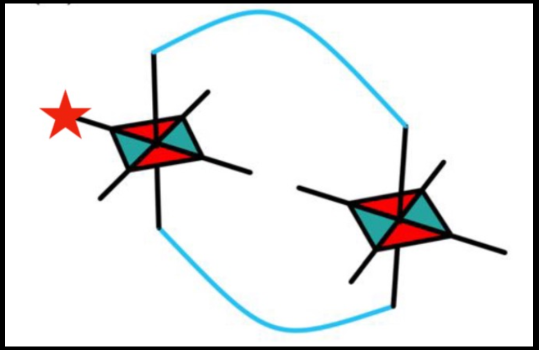
WHAT CAN YOU DO WITH THIS?

- Not obvious how useful the codes you can construct this way…

WHAT CAN YOU DO WITH THIS?

- Not obvious how useful the codes you can construct this way…

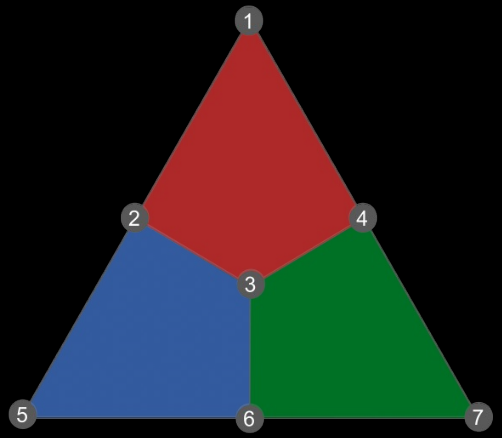
Steane [[7,1,3]]



WHAT CAN YOU DO WITH THIS?

- Not obvious how useful the codes you can construct this way…

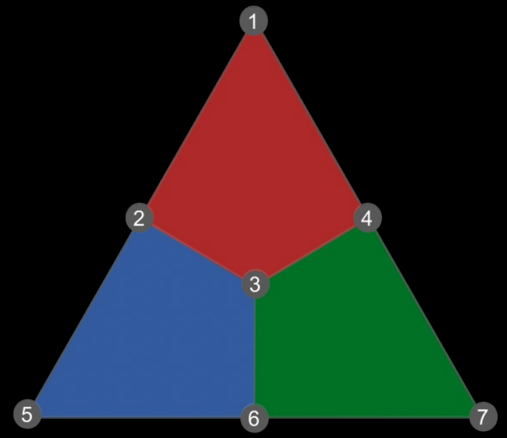
Steane [[7,1,3]]



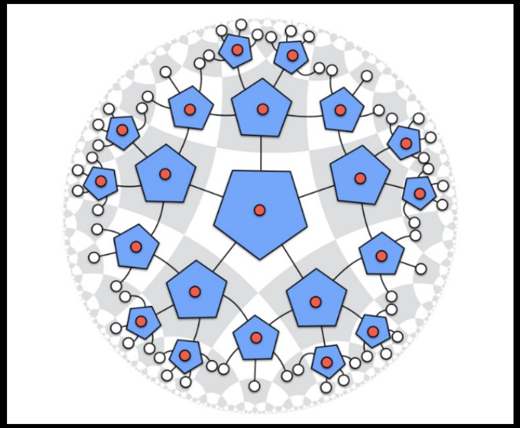
WHAT CAN YOU DO WITH THIS?

- Not obvious how useful the codes you can construct this way…

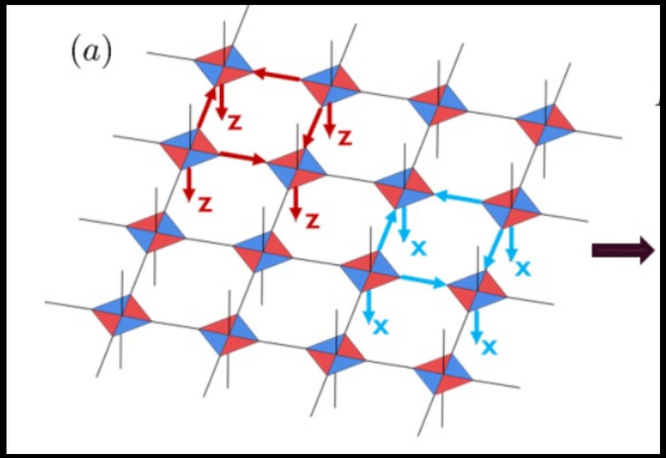
Steane [[7,1,3]]



HaPPY Code



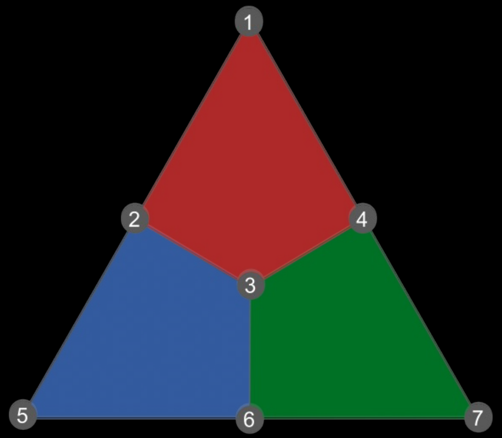
Toric Code



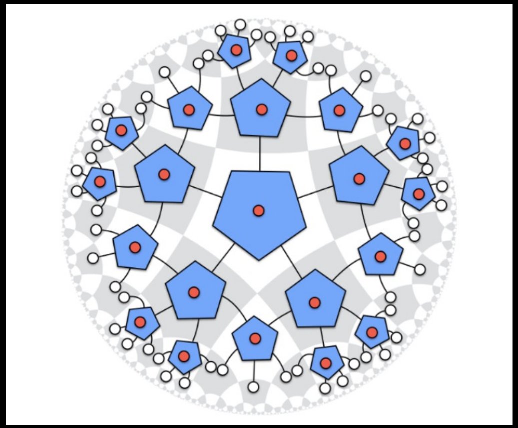
WHAT CAN YOU DO WITH THIS?

- Not obvious how useful the codes you can construct this way…

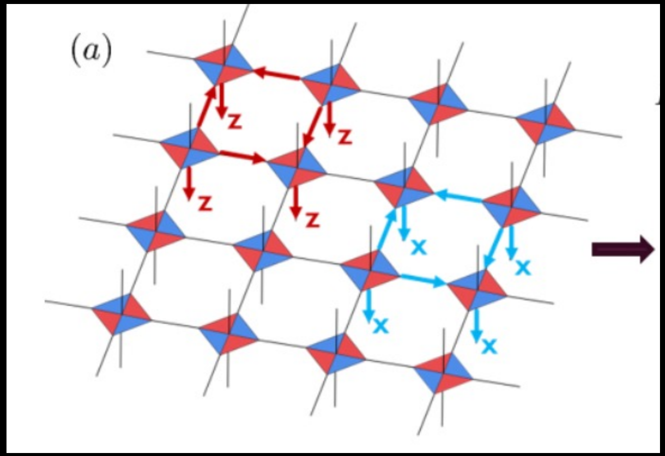
Steane [[7,1,3]]



HaPPY Code*



Toric Code



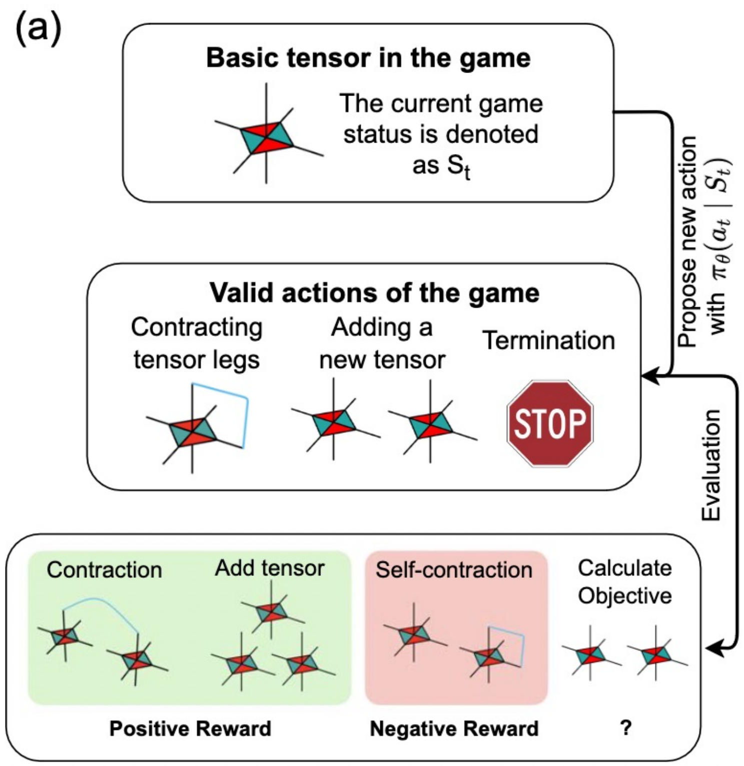
- These are all achievable with many copies of the **same** lego!

LET'S PLAY A GAME

- The construction so far is tedious but simple (operator matching + pushing)
- Can an RL agent learn to produce new codes?
- Need to specify the ingredients

	Chess	**Lego**
States	Board configurations	Lego pieces, leg contractions
Actions	Move, capture, castle	Add a tensor, connect legs, terminate
Reward	Checkmate, capture pieces, …	**????????**

GAME OVERVIEW



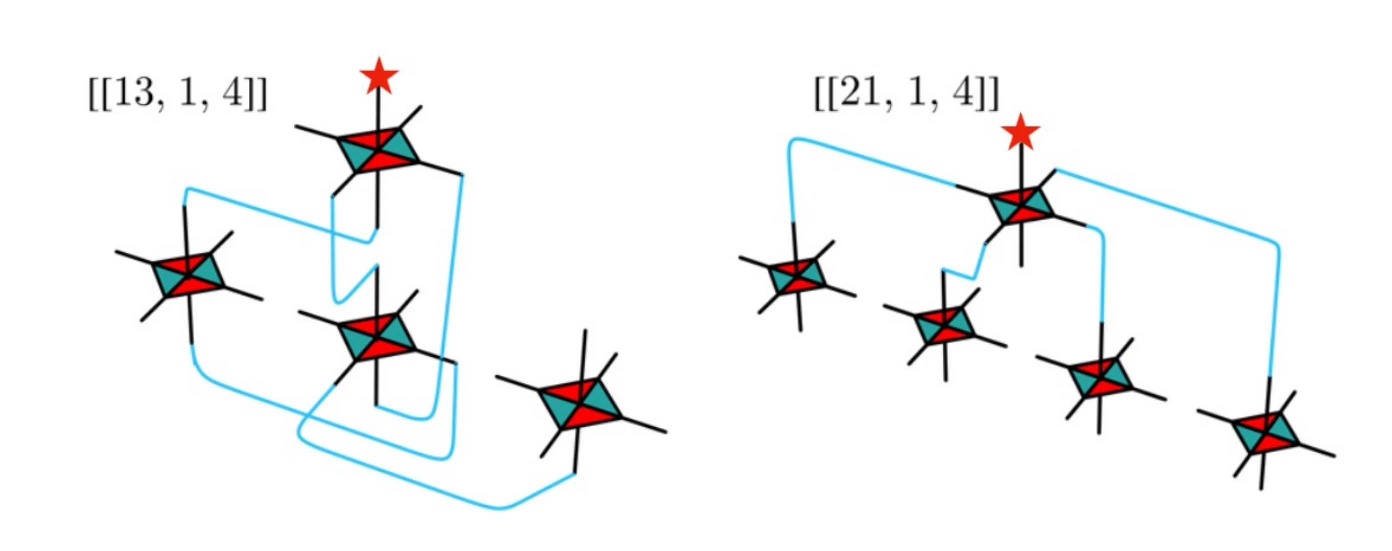
GAME: DISTANCE MAXIMIZATION

- The code distance *d* is a measure of robustness to adversarial errors
  - Counts minimum weight of logical operators
- Task: given a handful of T6 legos, produce a high distance code

GAME: DISTANCE MAXIMIZATION

- The code distance *d* is a measure of robustness to adversarial errors
  - Counts minimum weight of logical operators
- Task: given a handful of T6 legos, produce a high distance code

Learned code — Naïve Code Concatenation



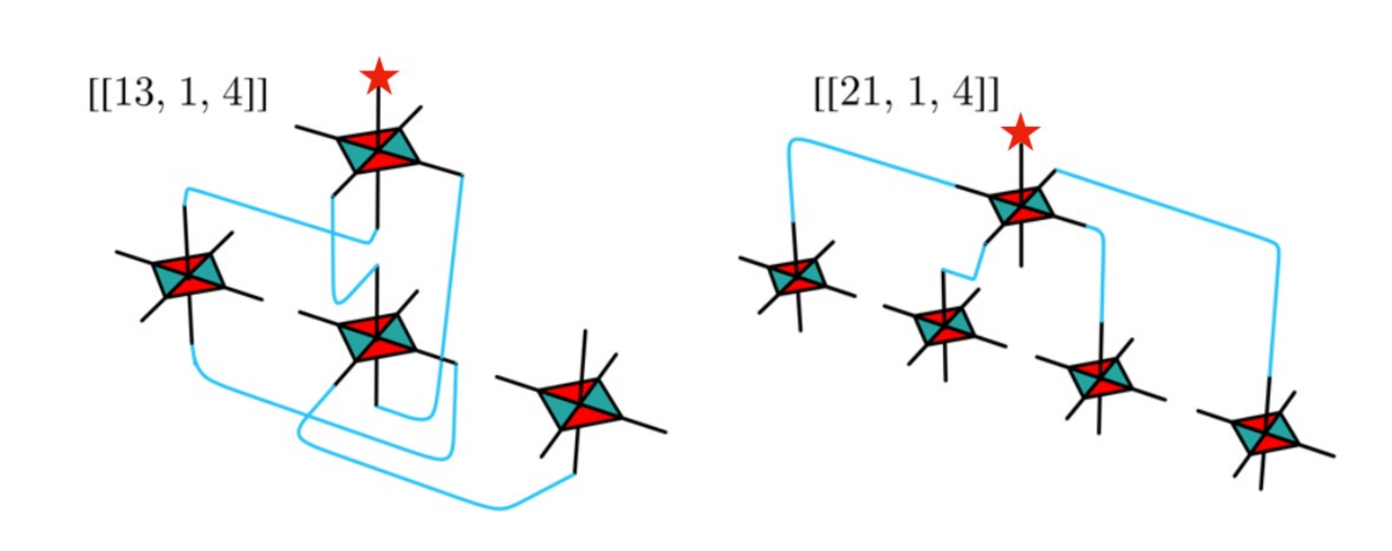
GAME: DISTANCE MAXIMIZATION

- The code distance *d* is a measure of robustness to adversarial errors
  - Counts minimum weight of logical operators
- Task: given a handful of T6 legos, produce a high distance code

Learned code

Naïve Code Concatenation

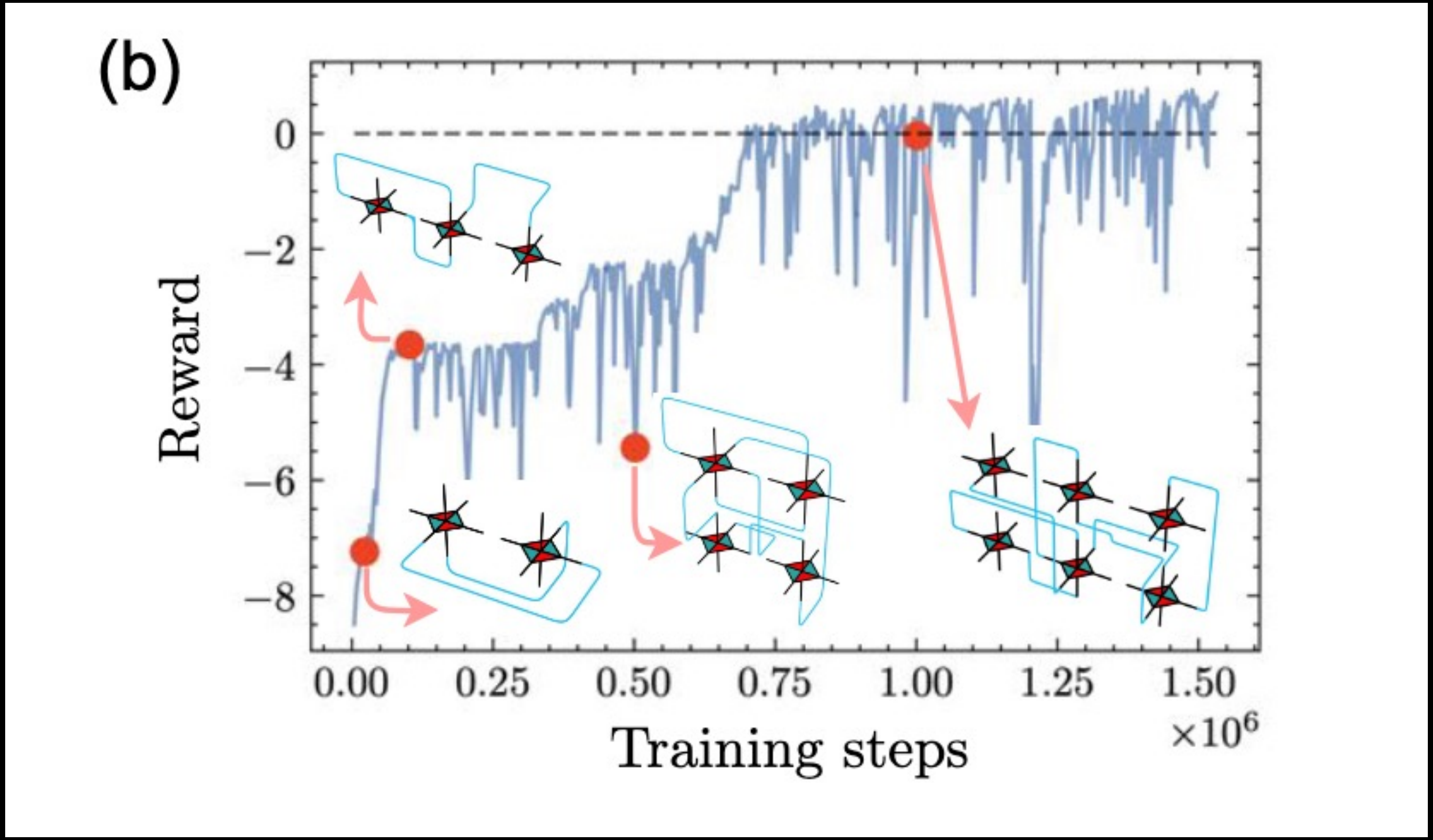
Saturates LP bound for CSS codes!



GAME: PROTECT AGAINST BIASED NOISE

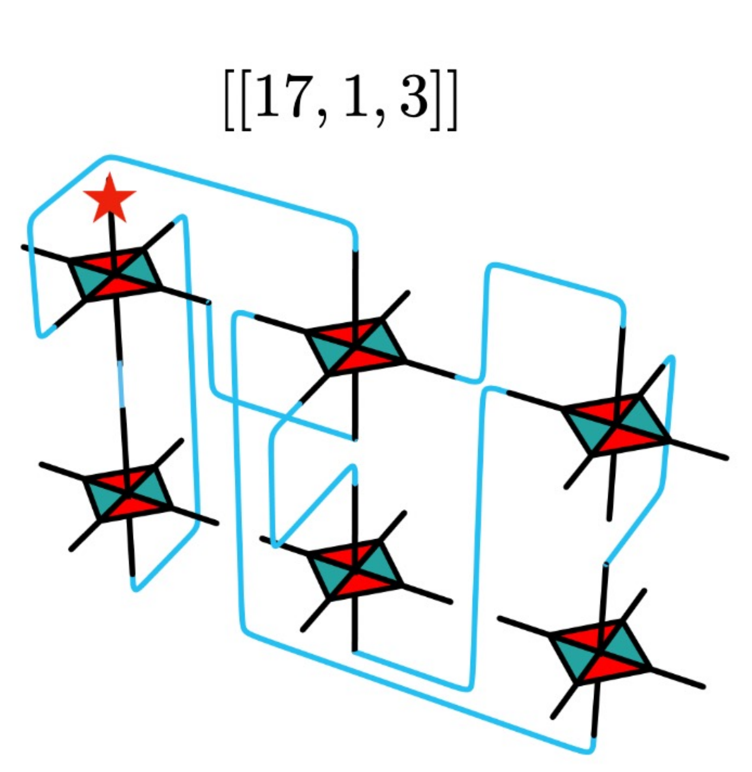
- Suppose you have a QC where Z flips occur more often than X flips.
  - $p_z = 0.05, p_x = 0.01$ independently for each qubit
- A logical error occurs if errors accumulate to become a logical operation
  - e.g. with probability $(p_x)^2(1-p_x)^2(1-p_z)^4$ we accidentally apply X_3X_4
- Task: Minimize the probability of a logical error for a single qubit
  - Under this model, a 20 qubit surface code has 4.38 x 10^{-6} chance of a logical error occurring

OVERVIEW OF LEARNING



GAME: PROTECT AGAINST BIASED NOISE

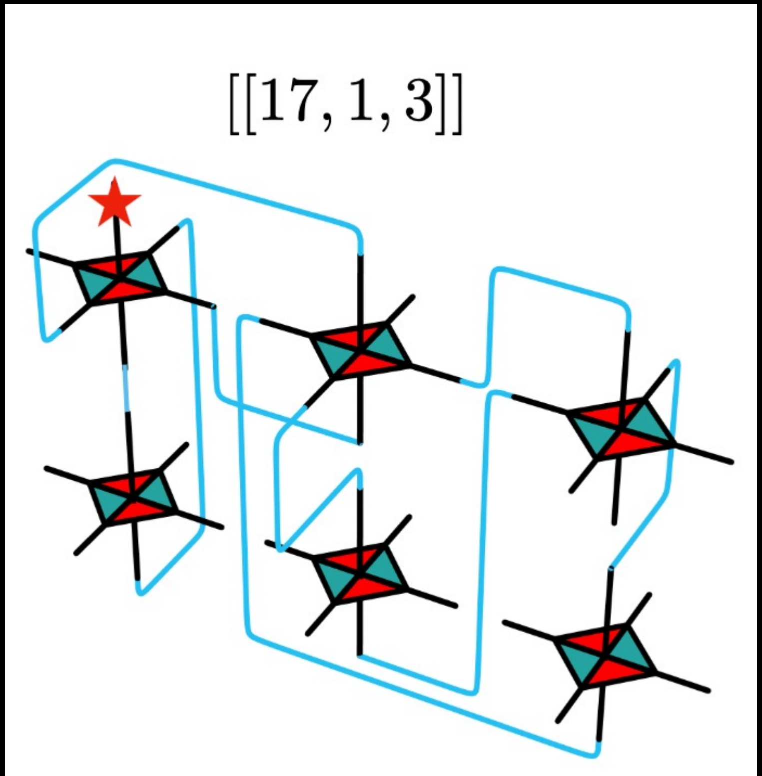
Using the exact same machinery, but tailoring the reward, we get the following code



Code	$[[n, k, d_X/d_Z]]$	p_L (10^{-5})
T6 BN 13A	$[[13, 1, 3/4]]$	.973
T6 BN 13B	$[[13, 1, 3/4]]$	.973
CSS Self-Dual†	$[[13, 1, 3/3]]$	26.8
T6 DM 13†	$[[13, 1, 4/4]]$	5.68
Reed-Muller	$[[15, 1, 3/7]]$	1.43
Surface (4x4)	$[[16, 1, 4/4]]$	1.46
XZZX (4x4)	$[[16, 1, 4/4]]$	1.07
T6 BN 17	$[[17, 1, 3/4]]$	**.404**
CSS self-dual†	$[[17, 1, 5/5]]$	.726
2D Color	$[[19, 1, 5/5]]$	.456
XZZX (4x5)	$[[20, 1, 4/4]]$	.665
XZZX (5x4)	$[[20, 1, 4/4]]$	.665
Surface (4x5)	$[[20, 1, 4/5]]$	.438
Surface (5x4)	$[[20, 1, 5/4]]$	6.58

GAME: PROTECT AGAINST BIASED NOISE

Using the exact same machinery, but tailoring the reward, we get the following code



Code	$[[n, k, d_X/d_Z]]$	p_L (10^{-5})
T6 BN 13A	$[[13, 1, 3/4]]$	.973
T6 BN 13B	$[[13, 1, 3/4]]$	.973
CSS Self-Dual†	$[[13, 1, 3/3]]$	26.8
T6 DM 13†	$[[13, 1, 4/4]]$	5.68
Reed-Muller	$[[15, 1, 3/7]]$	1.43
Surface (4x4)	$[[16, 1, 4/4]]$	1.46
XZZX (4x4)	$[[16, 1, 4/4]]$	1.07
T6 BN 17	$[[17, 1, 3/4]]$	**.404**
CSS self-dual†	$[[17, 1, 5/5]]$	.726
2D Color	$[[19, 1, 5/5]]$	.456
XZZX (4x5)	$[[20, 1, 4/4]]$	.665
XZZX (5x4)	$[[20, 1, 4/4]]$	.665
Surface (4x5)	$[[20, 1, 4/5]]$	.438
Surface (5x4)	$[[20, 1, 5/4]]$	6.58

Distance is lower!

TAKEAWAYS

- Quantum Legos provide a modular toolkit for building QECCs
- We discovered some cool codes!
  - Saturate linear programming bounds on distance for CSS codes
  - Beating 2D surface code variants at i.i.d. biased Pauli noise
- Designing new codes is a difficult, ambiguous task
- We propose a framework for discovering new codes that can be tailored to any purpose
  - Sometimes, yields counter-intuitive or surprising results

FUTURE DIRECTIONS

- Different lego blocks?

- Simulation/experiment hybrid game?

- What code properties do we really want to optimize for?
  - Encoding rate
  - Device specific error models
  - Low weight check operators
  - Hardware/connectivity constraints

- Reverse engineering the HaPPY, toric code?

THANK YOU

ROBUSTNESS OF CODE

Outperforms over a range of p_z, p_x