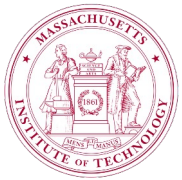
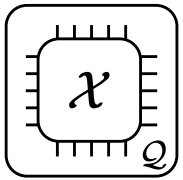
Challenges in Scalable Quantum Error Correction

... or things that keep us up at night

Zhiyang He (Sunny), MIT



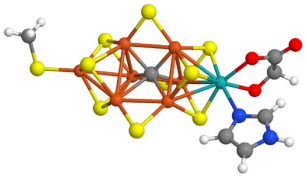


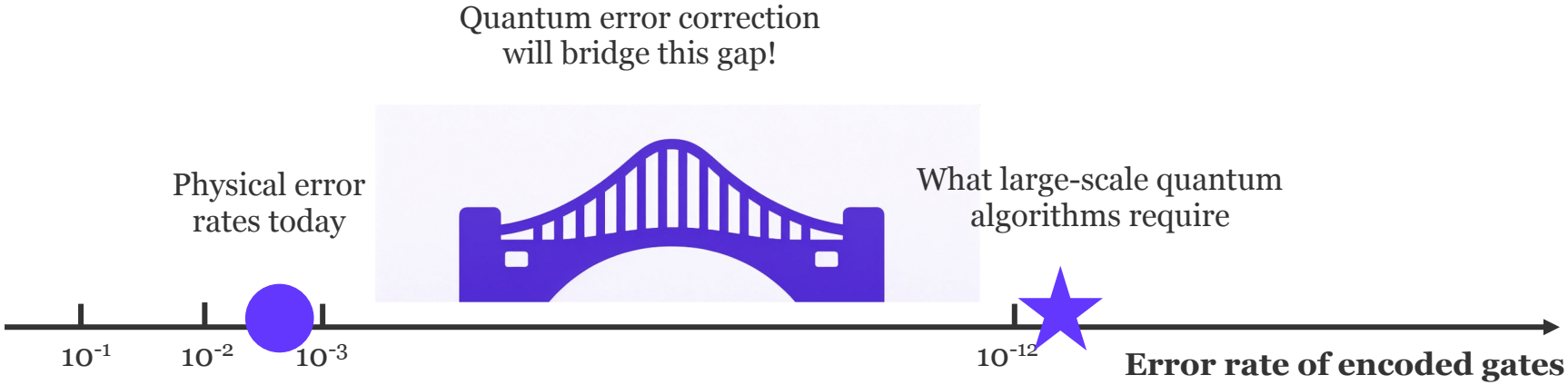
The Promise of Quantum Error Correction

We want to run large quantum algorithms, but how large?

- Circuit volume = Width (space) × Depth (time), V = WD.
- Fault-tolerant execution → gate error rate at $\Theta(1/V)$.
- Factoring 2048-bit: around 10^{15} qubit-steps.

$n = p \cdot q$

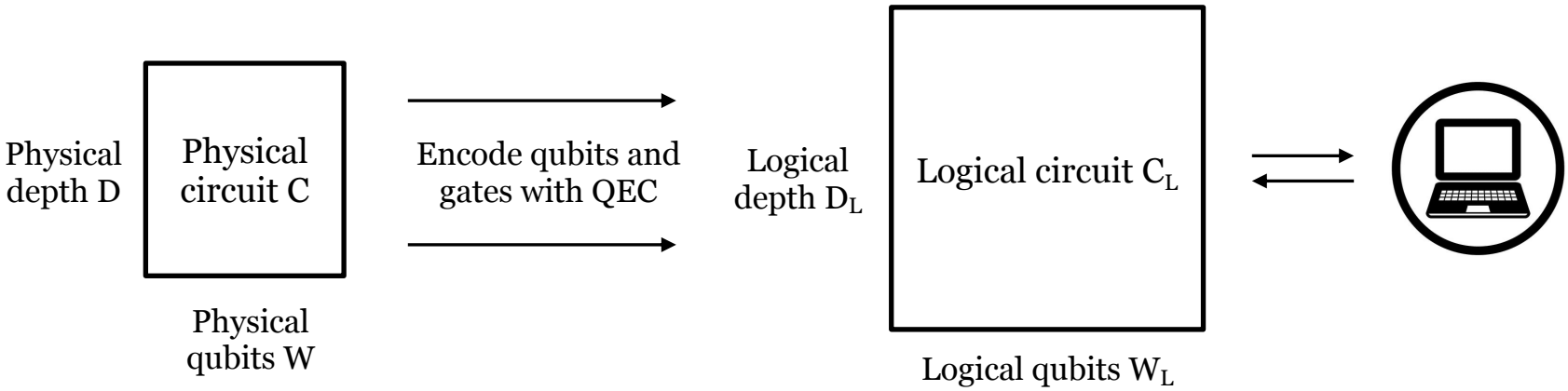




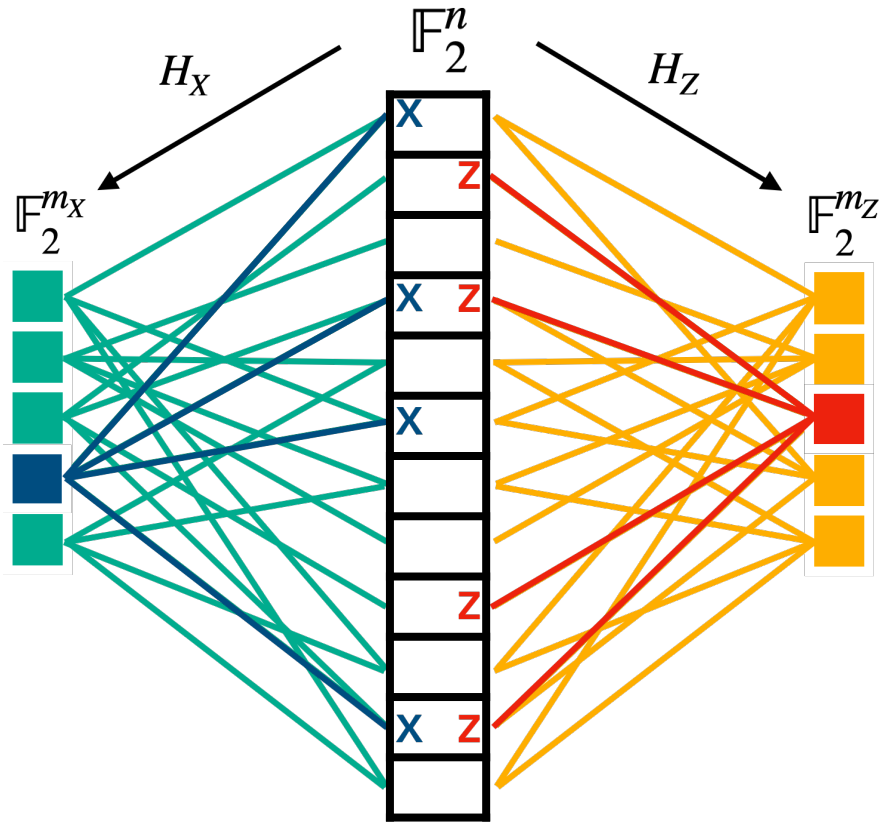
FTQC: Fault-Tolerant Quantum Computation

FTQC: Encode physical qubits/gates/error rates into logical qubits/gates/error rates.

- Space overhead: W_L/W, Time overhead: D_L/D.
- Often measured in terms of physical circuit volume V, specifically log(V).



QEC Basics: CSS Codes



Quantum CSS codes:

- Two binary parity check matrices H_X and H_Z. So called X- and Z-checks.
- Duality: $H_X H_Z^T = 0$.
- Distance d = minimum weight of certain linear subspaces.

Logical error rate often scales at $p_L = e^{-\Theta(d)}$.

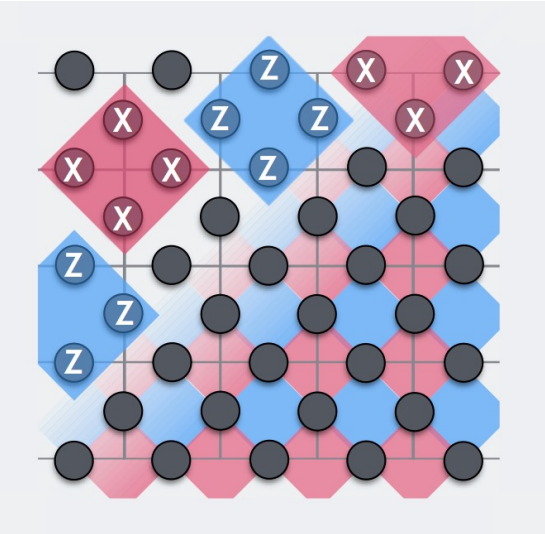
- Set $d = O(\log V)$. Determines spacetime overhead of FTQC.

* Figure cred. Louis Golowich.

From Topology to Combinatorics

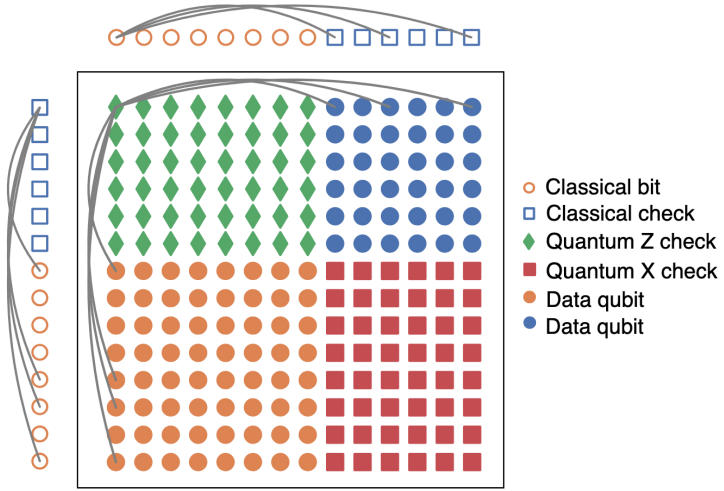
Surface code:

- [[n, k, d]] = [[$\Theta(d^2)$, 1, d]].
- Space overhead: $\Theta(d^2) \approx \log^2 V$.
- Weight 4 checks on grid connectivity → great for hardware!



Quantum low-density parity-check (LDPC) codes

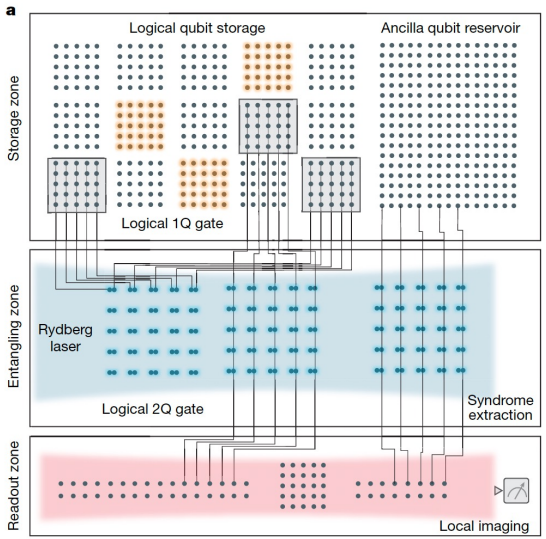
- Constant weight checks on unrestrained connectivity
- Can have k, d = $\Theta(n)$.
- Constant rate → space overhead = $\Theta(1)$!



Homological Product (HGP) Codes

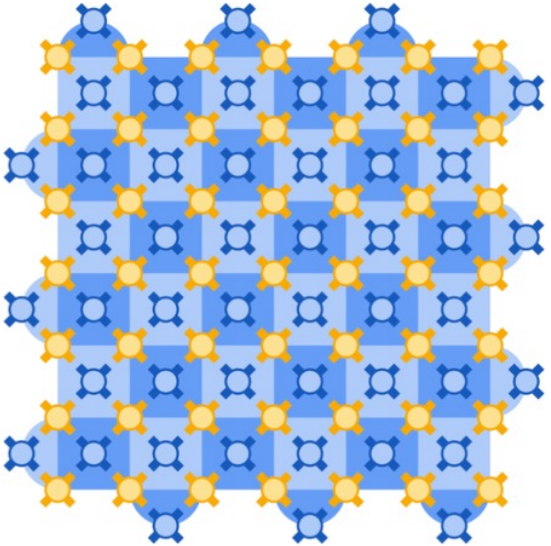
* Figure cred. Niel de Beaudrap and [Xu et al. 2308.08648].

Building a Quantum Computer in 2025



Harvard

IBM

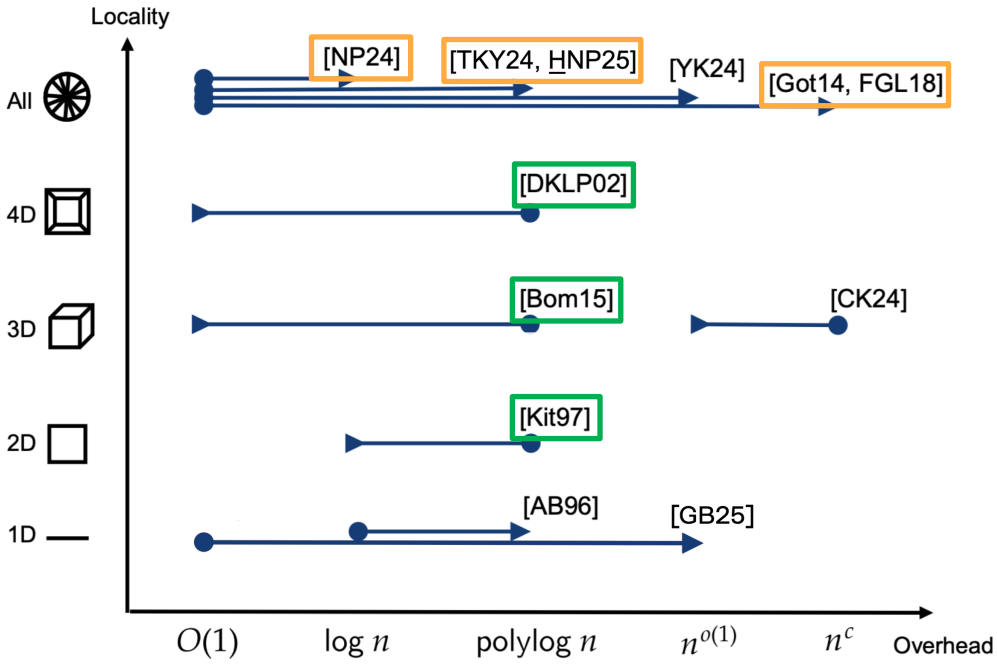


Google

- Fast hardware progress enabling long range connectivity across many platforms.
- Space is more constraining than time, LDPC codes widely studied
- Theme of our time: Codesign hardware, error correcting system, and applications.

I. Asymptotic Overheads and Barriers

Saving space… but at what cost?



Triangle marks time overhead, circle marks space overhead.
Assumption on classical computational resources vary.

Topological codes, which encode one qubit per block, incur polylog space overhead, but can get constant time overhead.

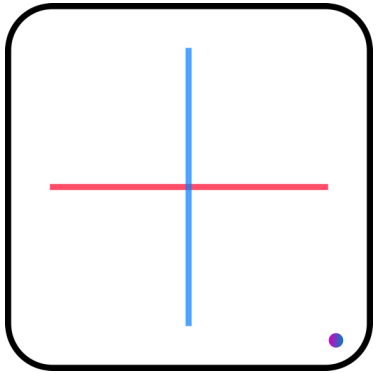
QLDPC codes, which encode many qubits per block, can get constant space overhead, but incurs polylog time overhead.

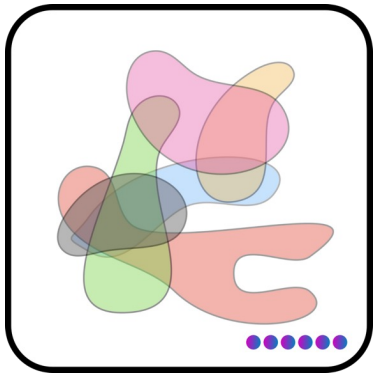
* Figure modified from [Gidney, Bergamaschi 2502.16132].

The Addressability Challenge

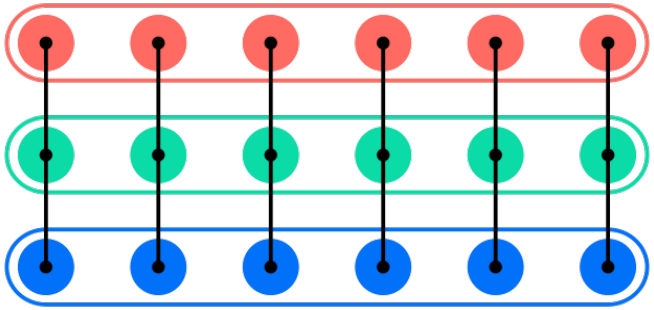
A (fundamental?) challenge: on high-rate memory, it is hard to perform addressable logic.

- Global physical action often induces global logical action.
- Non-global physical action often breaks code space.
- It is hard to act on selected set of logical qubits, or just single logical qubits.
- As a result, given a depth-1 logical circuit on k qubits, we often need to compile it into O(k) different encoded operations, which executes sequentially.
- When $k = O(d) = \log(V)$, incurs a log(V) time overhead.

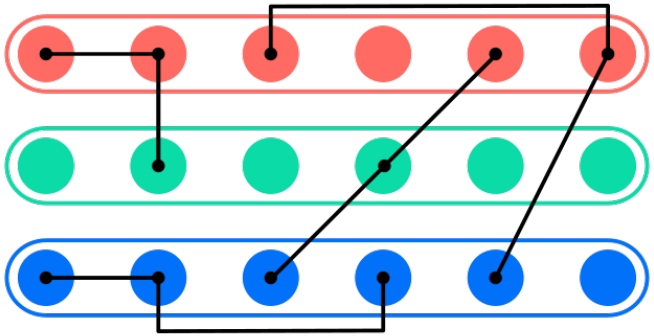




Partial Progress: Addressable CCZ



Global: must act on all triples of logical qubits



Addressable: can act on *an* arbitrary triple

In [2502.01864] and [2507.05392], HVWZ constructed asymptotically good codes with addressable CCZ gates.

- Asymptotically good: $k, d = \Theta(n)$
- CCZ: a powerful 3-qubit non-Clifford gate
- Addressable CCZ: different from global CCZ known in prior works*

Good progress, but not enough addressability.

- A depth-1 logical CCZ circuit still need a depth-O(k) physical circuit.

* [Wills, Hsieh, Yamasaki 2408.07764]; [Nguyen 2408.10140]; [Golowich, Guruswami 2408.09254].

Open problems

Can we derive a $\log(V)$ lower bound for the spacetime overhead of FTQC?

- Assumption on classical computation varies: usually ignore classical space overhead but account for classical time overhead.

Can we formalize the tension between code parameters and addressability of gates?

- Upper bound on parameter given addressable gates, or vice versa.
- Recent progress by [Krishna and Zémor 2510.03057], also GJ25.**

II. QLDPC Codes in Practice

Fast Progress in QLDPC Memory

Recent constructions, [n, k, d]:

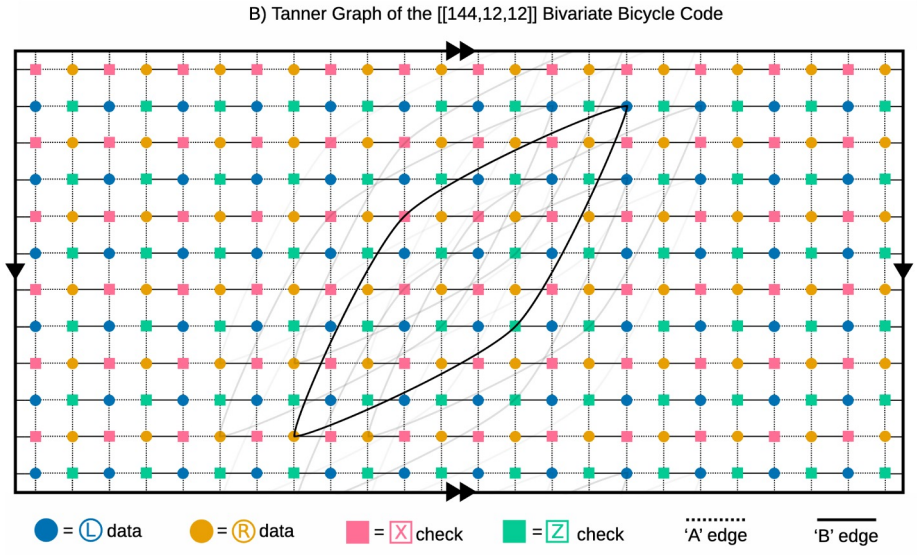
- Bivariate Bicycle code [144, 12, 12] *
- Homological product code [2500, 100, 12] **
- Lifted/balanced product code [544, 80, ≤ 12] **

Surface code: [265, 1, 12].

Many more finite size code constructions now.

Memory: Decoding algorithm, threshold and logical error rate, hardware…

➢ Lots of progress, lots to be done.



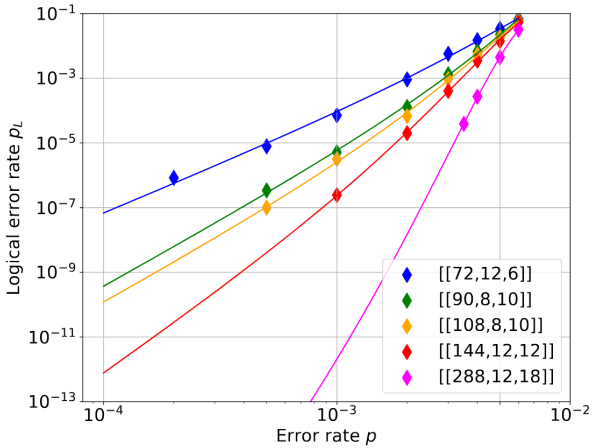
* [IBM Team 2308.07915]. ** [Xu et al. 2308.08648].

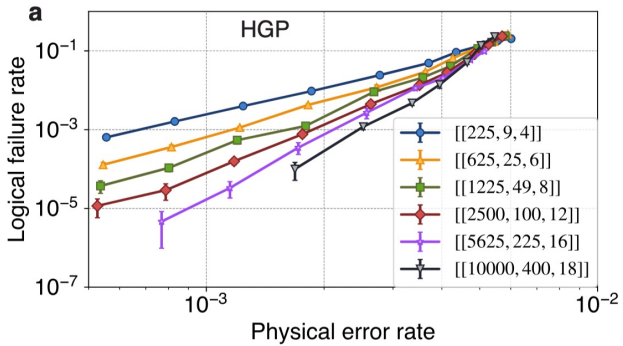
Scaling to Larger Distances

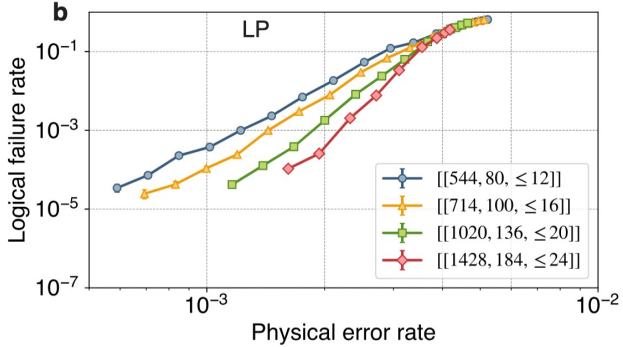
The bar for good memories is ever-rising. We want to find codes with good rate and higher distances.

Challenges:

1. At medium scale, only one known construction gives constant rate and provable, growing distance: homological product (HGP) codes.
2. Verifying distance is otherwise computationally hard.







Open problems

Code constructions, decoders and distance/logical error estimation are all exciting areas of research. Ask us about them!

Can we get random quantum LDPC codes with nice parameters?

- Let's set a small target for ourselves, maybe start with finding a asymptotically good family?
- Naïve approach fails due to the duality requirement: $H_X H_Z^T = 0$. I.e., the two classical codes need to have constant weight subspace.
- In practice: can we sample from popular families, such as generalized bicycle codes, with provable guarantees?

From Memories to Computers

Performing encoded computation on QLDPC codes is a *very* active research direction.

> Performing logical computation on QLDPC memory has been a long standing challenge in theory and in practice, with extensive research proposing many schemes [BB24, BCG+24, QWV23, ES24, ZSP+23, SPW24, BDET24, HKZ24, Lin24, GL24, MGF+25, BMD09, VB22, BVC+17, LB18, JO19, KP21, CKBB22, SKW+24, CB24, Cow24, CHRY24, WY24, SJOY24, IGND24, ZL24, CHWY25, HJOY23, XZZ+24, BGH+25, HCWY25, YSR+25]. And 5+ papers this week.

Usual style of a paper: On ___ codes, we implement ____ gates with ___ overheads.

- Commonly studied codes include BB and homological product codes.
- Most existing schemes have one or more of the following limitations:

a) Only works for specific code families;
b) Implements a small set of gates on the k-qubit logical space;
c) Incurs a heavy space/time overhead.

Putting them together for universal FTQC is like a jigsaw puzzle, except the pieces all have different shapes and sizes.

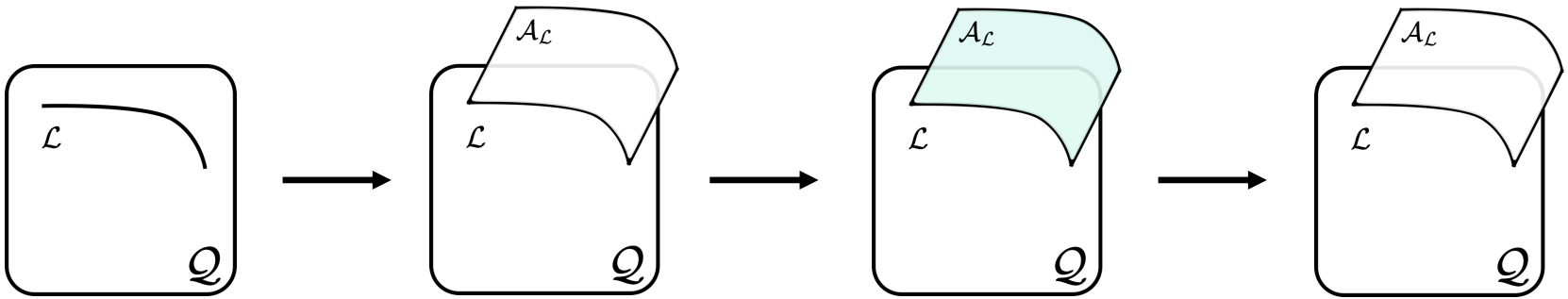
Code Surgery

Consider all 4^k Pauli operators L, and the corresponding rotations $\exp\{-i\frac{\pi}{4}L\}$, $\exp\{-i\frac{\pi}{8}L\}$.

- A large set of logical operations that are universal.

On *any* code, fix *any* L, the rotation $\exp\{-i\frac{\pi}{4}L\}$ can be implemented using a set of ancilla qubits.

- $\tilde{O}(d)$ additive space overhead, O(d) time overhead.
- Can perform $\exp\{-i\frac{\pi}{8}L\}$ when supplied with magic states → universal FTQC.



(a) Start: Code $\mathcal{Q}$ and operator $\mathcal{L}$.

(b) Init: Initialize ancilla in unentangled state.

(c) Merge: Switch to a new code by measuring new checks

(d) Split: Measure out ancilla and return to $\mathcal{Q}$.

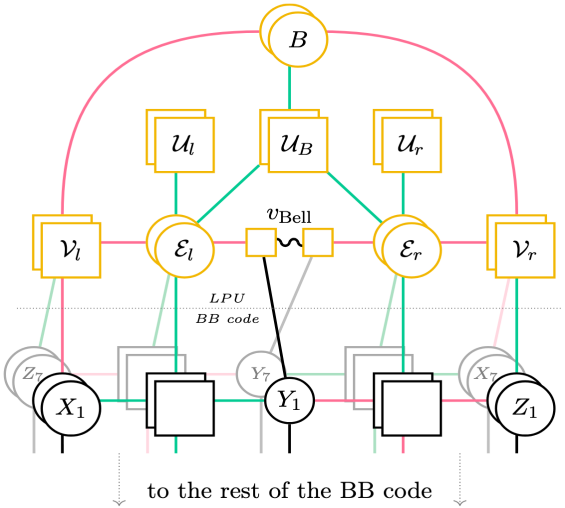
Logical Processing in BB Code

Code surgery has experienced lots of recent developments*, and is now the most promising way to perform FTQC with BB code.

[IBM Team 2506.03094] designed a Logical Processing Unit (LPU) for the [[144, 12, 12]] code

- Can perform 15 different rotations;
- Generate the full Clifford group on k qubits when combined with simpler global gates;
- Uses 90 ancilla qubits.

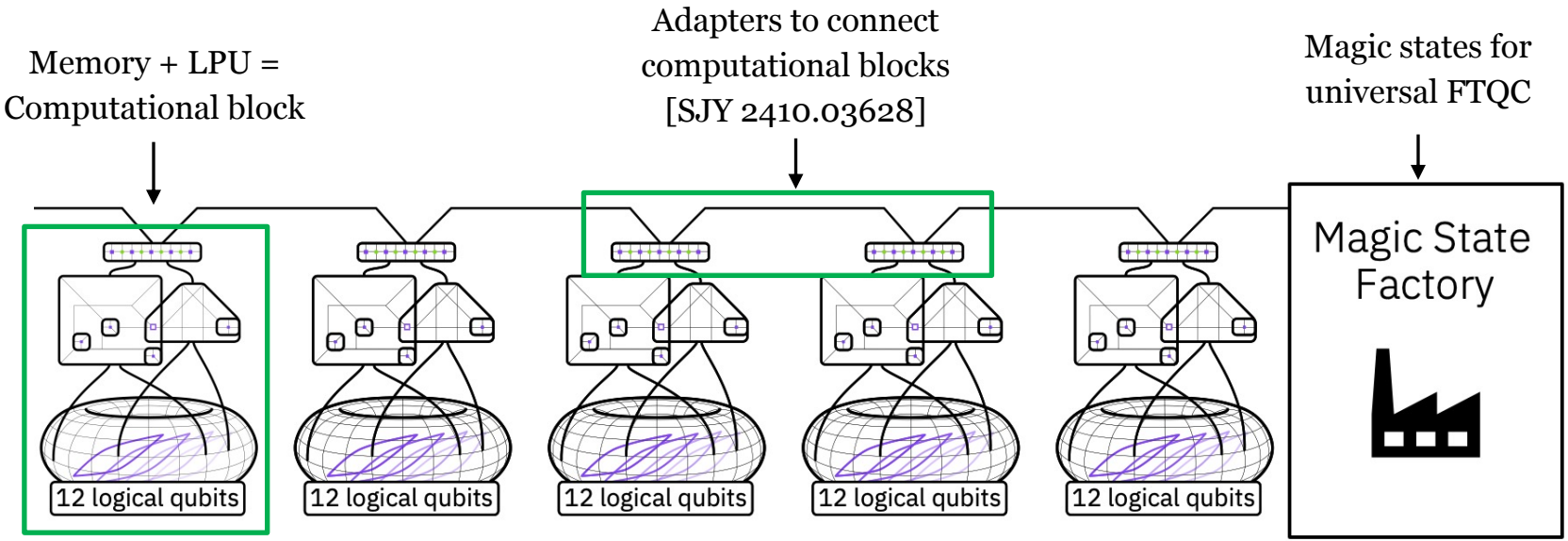
Improves older designs from [CHRY 2407.18393] and [WY 2410.02213].



Current design of a Logical Processing Unit (LPU) for the 144 BB code.

* [CKBB 22], [CHRY 24], [WY 24], [SJY 24], [ZL 24], [CHWY 25], [HCWY 25], [IBM Team 25], [BBC 25], more to appear.

Bicycle Architecture



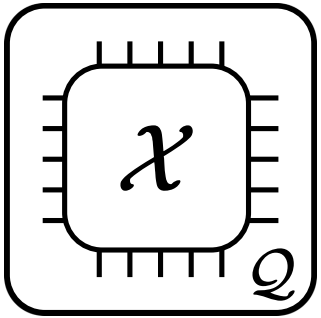
Bicycle architecture, 2506.03094.

Generalization: Extractor Architecture

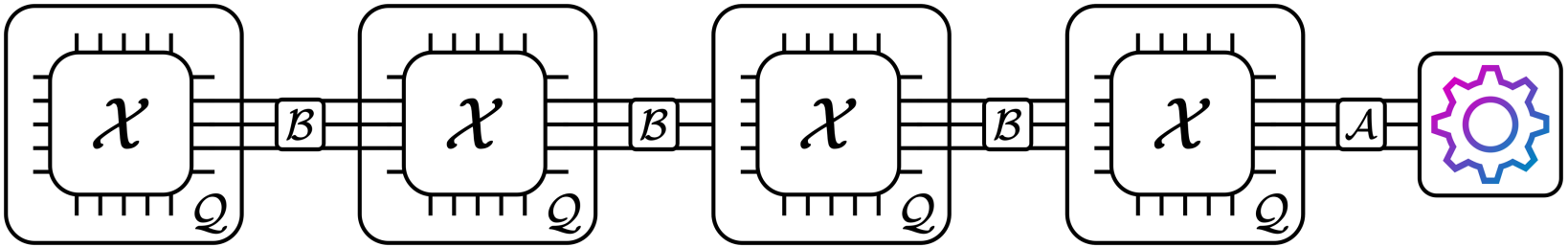
Theoretical result by [HCWY 2503.10390].

- For any quantum code Q, we can build an extractor X, which can execute all 2×4^k rotations (when supplied with magic states).
- I.e., full processing of the logical space enabled by one ancilla system.
- Size $\tilde{O}(n)$, expect to be $O(n)$ in practice.

Enables extractor architectures: universal FTQC in low overhead that can be built with *arbitrary* code and magic state supply.



Extractor Augmented Computational (EAC) Block

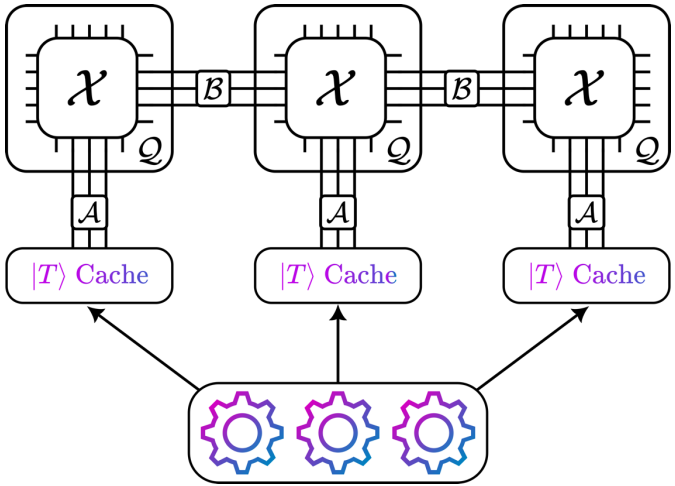


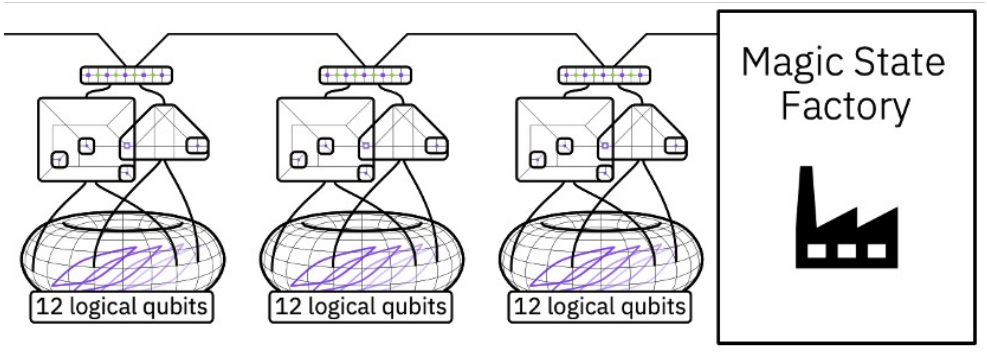
A Promising Blueprint

Distinctive features of bicycle and extractor architectures:

- Can be built with fixed, constant-degree physical qubit connectivity.
- Compilation removes most (or all) logical Clifford operations.
- Good asymptotic performance *and* practical performance at thousand-qubit scale.

This is how we now envision a large-scale, fault-tolerant quantum computer.





Open problems

Lots of exciting and important work to be done for bicycle and extractor architectures. Ask us about them!

Our architectures are engines which consumes magic states as fuel. Can we produce high-fidelity magic states more effectively in QLDPC codes?

- Great asymptotic progress: [WHY 2408.07764], [NP 2411.03632].
- In practice, we are still looking for high-performing schemes.

III. Spacetime Tradeoff in Applications

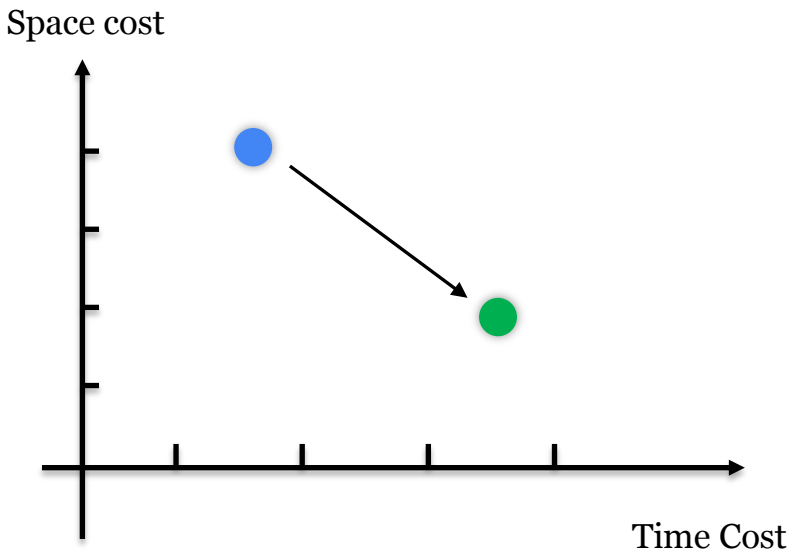
The Economy of Trading Time for Space

For a foreseeable while, space is more constraining than time. For an application, reducing qubit count (at the expense of depth) brings it closer to practical implementation.

E.g. Reducing qubit count by 2x while increasing depth by 10x:

- Circuit volume increased by 5x, but
- Space overhead for FT implementation reduced, because distance only increases additively.

Many exciting ideas here: catalytic computing, circuit cutting, new factoring algorithms…



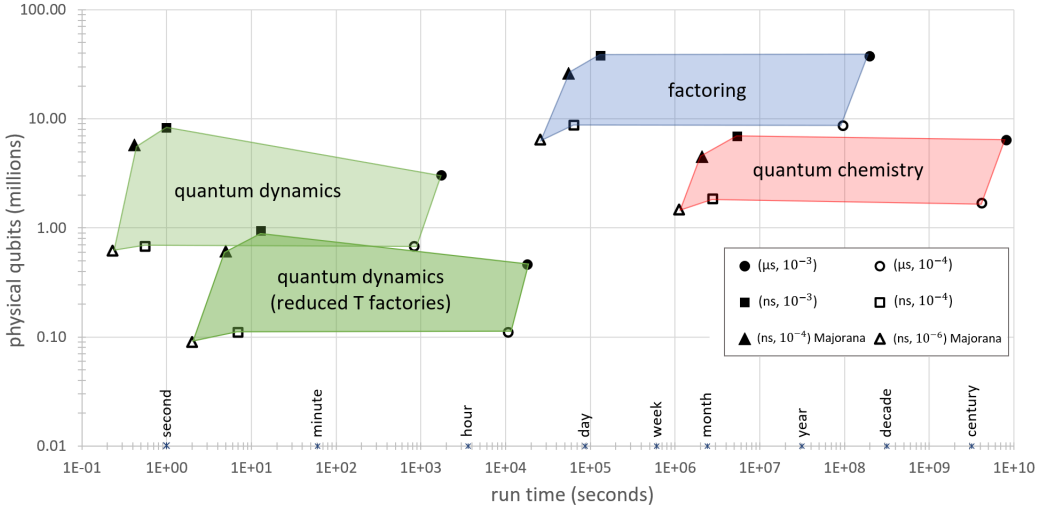
The Eventual Need for Speed

In the future regime of large scale FTQC:

- We cannot build more time, and
- The computing systems are likely modular.

For industrial applications, it would be worthwhile to explore:

- Reducing circuit depth by parallelism, such as Regev's Factoring.
- Implementation on modular architecture and trading space for time.



Resource estimate from Beverland et al. 2211.07629

Challenges in Scalable Quantum Error Correction

Zhiyang He (Sunny), MIT

Many thanks to Andrew Cross, Anirudh Krishna, Harry Zhou, Chris Pattison, Adam Wills and Katie Chang for help with designing this talk.