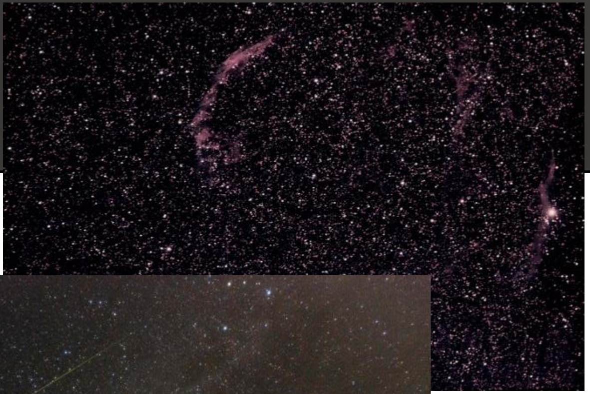
2024/08/31 AQIS2024 Satellite WS for FTQC

A computer system perspective of large-scale quantum computers

SOKENDAI / OIST / JSPS
Shin Nishio

Phys. Rev. A **107**, 032620
IEEE TQE **4 (2023): 1-7**
arXiv:**2406.08832**
arXiv:**2302.00267**

Shin Nishio



2022.4～
- QIST unit, supervised by Kae Nemoto

TQC2024(Sept9-13) is in OIST

Outline

1. Background and overview of large-scale quantum computation

2. System software
   Circuit optimization of logical circuit

3. Modular architecture
   1. Quantum multiplexing for interconnects
   2. Intermediate representation (Programming Language)

4. Towards large-scale quantum computer

1. Background: Motivation

P ⊆ BPP ⊆ **BQP** ⊆ PSPACE

- Prime factor (superpolynomial speedup)
  Classical : $\exp\left(\left((64/9)^{1/3}+o(1)\right)(\log n)^{1/3}(\log\log n)^{2/3}\right)$
  Shor[1]: $O(n^3)$

- Database searching (polynomial speedup)
  Classical: $O(n)$
  Grover[2]: $O(\sqrt{n})$

Discrete logarithms, quantum simulation, system of linear equation...

[1] Peter Shor "Algorithms for quantum computation: discrete logarithms and factoring." *Proceedings 35th annual symposium on foundations of computer science*. Ieee, 1994.
[2] Lov K. Grover “Quantum mechanics helps in searching for a needle in a haystack.” *Physical Review Letters*, 79(2):325-328, 1997.

Quantum Computing today

- Speed of atomic operations

High performance computing[3]

Fugaku(2020)

442 PFLOPS

clock rate for a processor 2.2 GHz

Quantum Computing (superconducting + optical interconnect)[4]

Single qubit gate 30 ns (33 MHz)

Two qubit gate 60 ns (16 MHz)

Measurement 240 ns (4 MHz)

Entanglement generation for remote nodes 1000 ns (1 MHz)

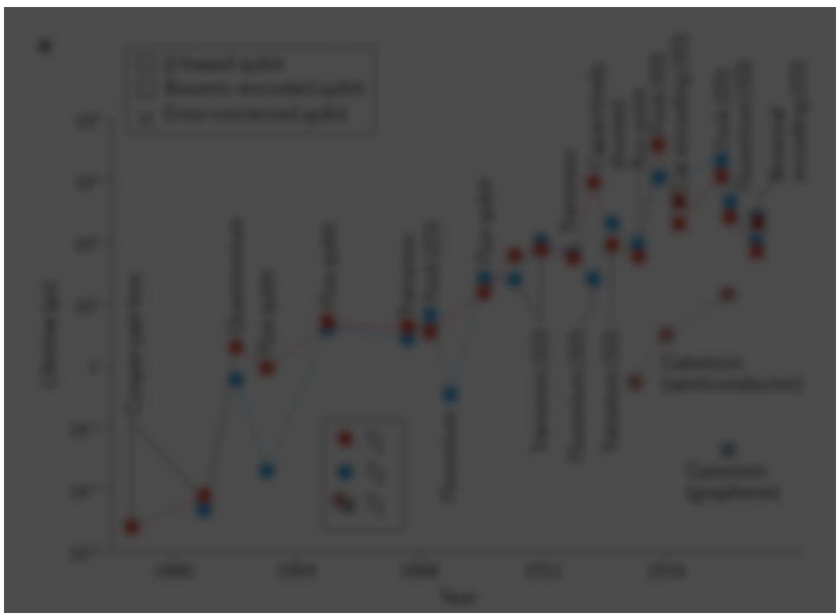
→$\mathbf{10^3}$ speedup is required!

[3] 安里彰 「コデザインによるスーパーコンピュータ富岳プロセッサの開発」電子情報通信学会 基礎・境界ソサイエティ Fundamentals Review (2022)
[4] Shin Nishio and Ryo Wakizaka. "InQuIR: Intermediate Representation for Interconnected Quantum Computers." arXiv preprint arXiv:2302.00267 (2023).

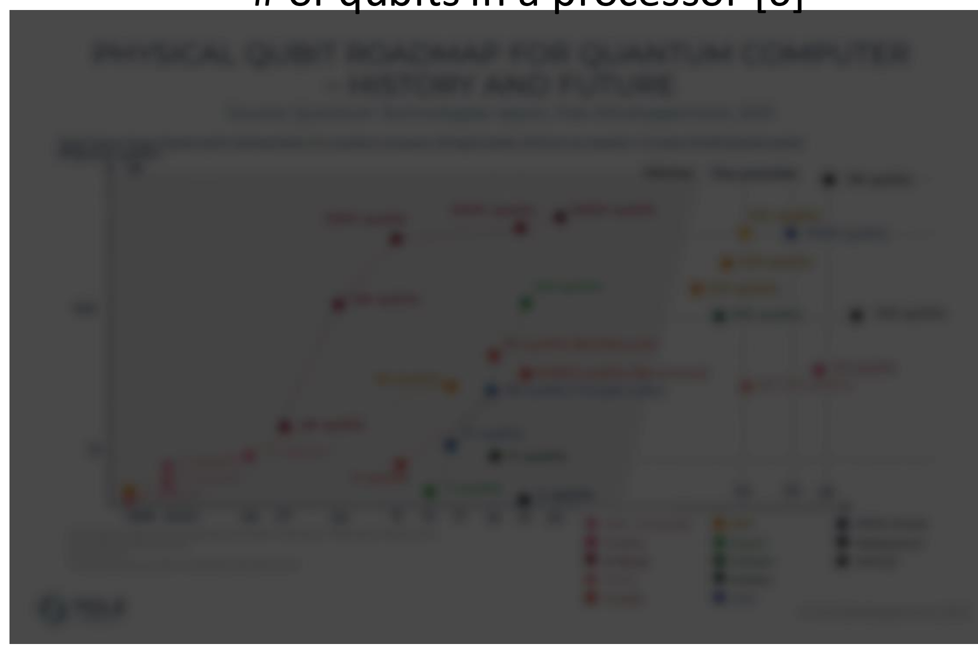
Fault-tolerant quantum computing

Qubit lifetime may saturate, but # of qubits is scaling
→ Fault-tolerant quantum computing may work

Lifetime of qubits [5]



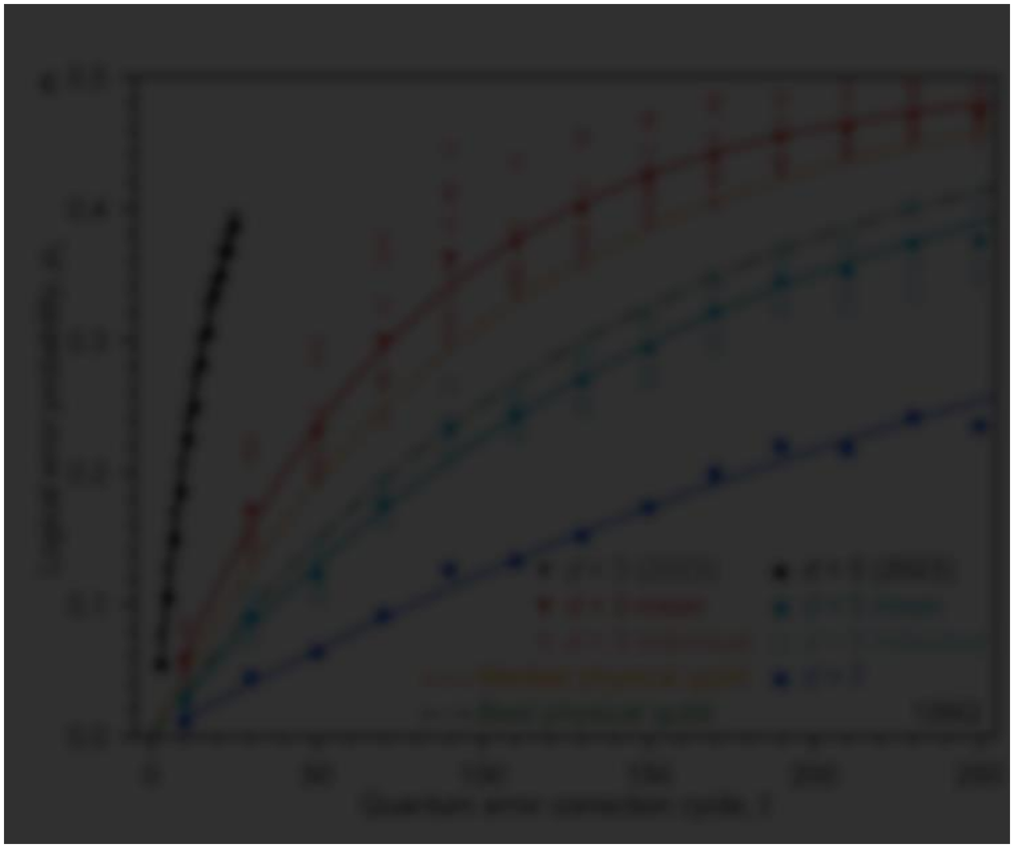
# of qubits in a processor [6]



[5] Irfan Siddiqi, "Engineering high-coherence superconducting qubits." *Nature Reviews Materials* 6.10 (2021): 875-891.
[6] Maurizio Di Paolo Emilio, "Current Status and Next in Quantum Computing." EETimes (2022)

Fault-tolerant quantum computing

Qubit lifetime may saturate, but # of qubits is scaling
→ Fault-tolerant quantum computing ~~may~~ work



From [7]

- Error suppression factor

$$\Lambda = 2.14 \pm 0.02$$

when increasing the code distance by two

- Real-time decoding

Latency of 63 μs at distance-5 up to a million cycles, with a cycle time of 1.1 μs

- Longer lifetime for logical qubit

2.4 ± 0.3 times better than the lifetime of physical qubits

[7] Google Quantum AI and Collaborators, Quantum error correction below the surface code threshold. arXiv:2408.13687

FTQC Overheads

Fault-tolerant Quantum Computing

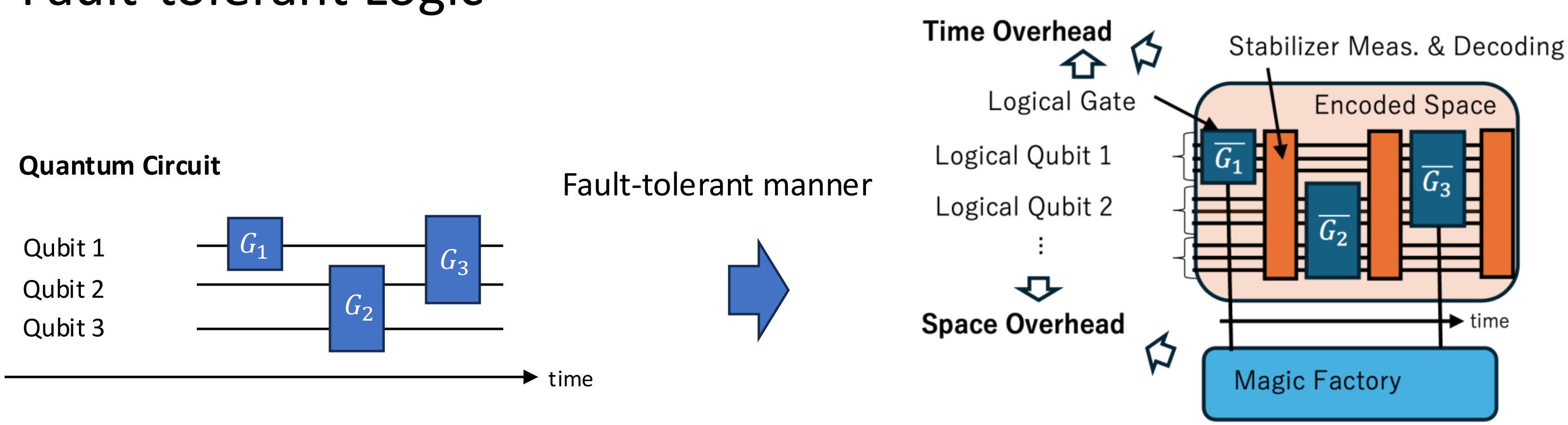
Space overhead (100k qubits)

- All the computation processes are on a $\boldsymbol{r}$ **times redundant code space** where $r = k/n$
- Cost of universality

  Magic state distillation (Tri-orthogonal codes, 3D color codes etc), code-switching, Concatenation, etc

Time overhead (days)

- Repetitive Syndrome measurement → Decoding → Correction
- Fault-tolerant Logic



Computational resources

logical

Quantum Algorithm

High-level quantum program
MCT, For, While, If, λ, Classical Computation

Solovay-Kitaev Compilation

(Discrete-gate) Quantum Circuit

Universal gate set {H, T, CX}

(Logical) Quantum Circuit

Lattice Surgery, Defect Braiding, Code-switching, Magic state distillation ...

Encoding

Decoding

Physical Quantum Circuit

Quantum Assembly-like Language

Physical Control

Physical

Program analysis and optimization

Language and decoding algorithms are core of quantum computing systems !

Computational resources

IEEE TQE **4 (2023): 1-7**

logical

Quantum Algorithm

High-level quantum program
MCT, For, While, If, λ, Classical Computation

Solovay-Kitaev Compilation

(Discrete-gate) Quantum Circuit

Universal gate set {H, T, CX}

(Logical) Quantum Circuit

Lattice Surgery, Defect Braiding, Code-switching, Magic state distillation ...

Encoding

Decoding

Physical Quantum Circuit

Quantum Assembly-like Language

Physical Control

Physical

Program analysis and optimization

Language and decoding algorithms are core of quantum computing systems !

2. System software

There are many good codes and its fault-tolerant logic, but some of them are not well-defined in low-layer

Formalize the circuit optimization rules which preserve unitary
↓
Check the computational complexity
↓
Find an easy case, efficient solver, approximation, etc.
↓
Introduce intermediate representation and then optimize

For reducing computational resources

What I can do as a computer scientist

0. Is the layer structure appropriate?

1. Find a bottleneck

2. Ease it

→ go back to 0

Logical Circuit

- Circuit optimization of logical circuit

Lattice surgery and defect braiding are candidates for the fault-tolerant logic for surface code computing

For the optimization problem of the logical quantum circuit,
- It is NP-hard for lattice surgery in general [8]
- The problem in a formal way has not been defined for defect braiding

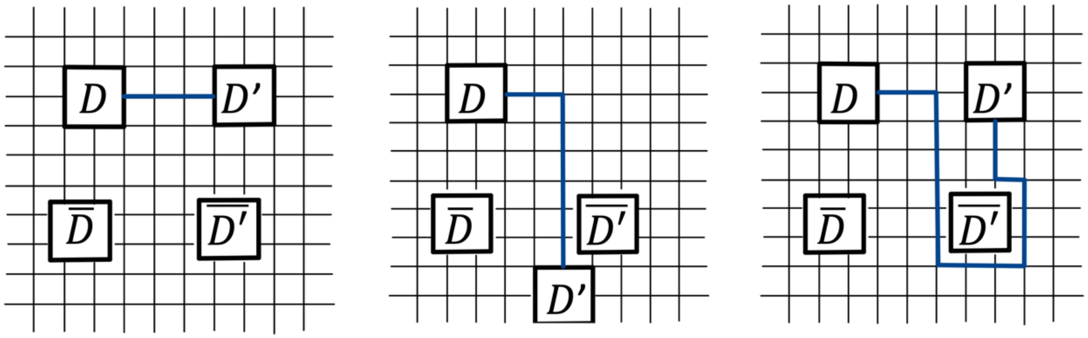
→ so we did!

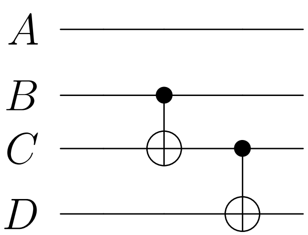
[8] Daniel Herr, Franco Nori, and Simon J. Devitt. "Optimization of lattice surgery is NP-hard." *Npj quantum information* 3.1 (2017): 35.
[9] K Wasa, S Nishio, K Suetsugu, M Hanks, A Stephens, Y Yokoi, K Nemoto, *Hardness of braided quantum circuit optimization in the surface code*, IEEE Transactions on Quantum Engineering 4, 1-7

IEEE TQE **4 (2023): 1-7**

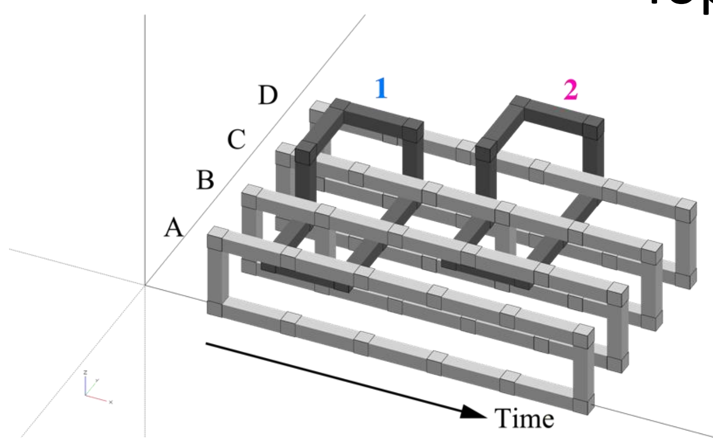
Computational complexity of DB circuit optimization

Defect braiding on surface codes realize two qubit logical interaction





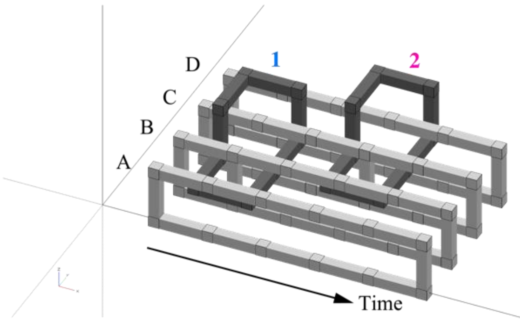
Topological circuit

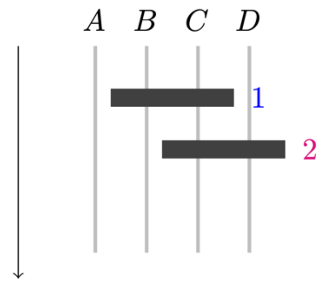


K Wasa, S Nishio, K Suetsugu, M Hanks, A Stephens, Y Yokoi, K Nemoto,
Hardness of braided quantum circuit optimization in the surface code,
IEEE Transactions on Quantum Engineering 4, 1-7

IEEE TQE **4 (2023): 1-7**

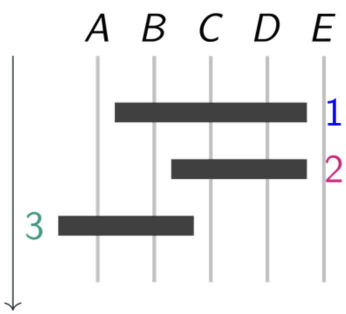
Computational complexity of DB circuit optimisation

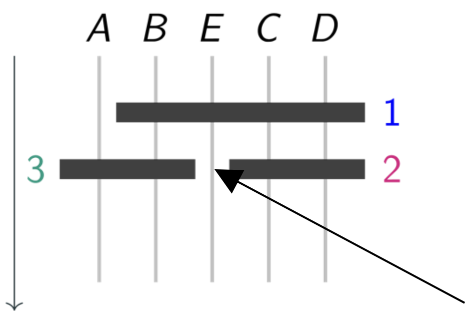




2D simplified circuit

Depth optimization with permutations of qubits (**Min-Braiding**)





$\pi: \{A, B, C, D, E\} \rightarrow \{A, B, E, C, D\}$

Enough margin between gates

Gates $\mathcal{R}$: $1 = \{B, C, D\}$ $2 = \{C, D\}$ $3 = \{A, B\}$

Partial order: $2 \succ 1, 3 \succ 1$

K Wasa, S Nishio, K Suetsugu, M Hanks, A Stephens, Y Yokoi, K Nemoto,
Hardness of braided quantum circuit optimization in the surface code,
IEEE Transactions on Quantum Engineering 4, 1-7

IEEE TQE **4 (2023): 1-7**

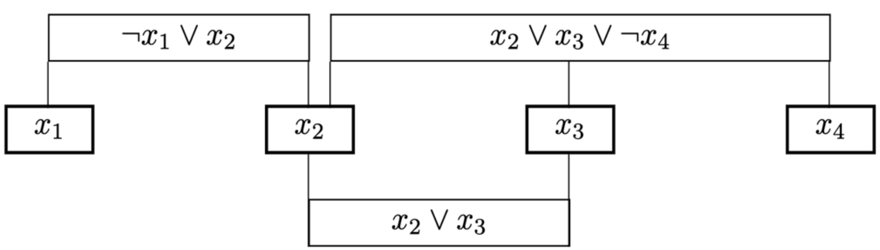
Computational complexity of DB circuit optimisation

We have proven that:

NP-completeness of Min-Braiding optimization

How?

- Assume that there exists an algorithm $\mathcal{A}$ that solves Min-Braiding efficiently.
- If it can be solved using $\mathcal{A}$ for any instance of the **known hard problem** $\mathcal{P}$, Min-Braiding is at least more difficult than $\mathcal{P}$.
  - We used **PlanarRectLinear 3SAT** (subset of 3SAT)



$$\phi = (\neg x_1 \vee x_2) \wedge (x_2 \vee x_3) \wedge (x_2 \vee x_3 \vee \neg x_4)$$
$$c_1 = (\neg x_1 \vee x_2), c_2 = (x_2 \vee x_3),$$
$$c_3 = (x_2 \vee x_3 \vee \neg x_4)$$
$$(x_1, x_2, x_3, x_4) = (0,0,1,0)$$

K Wasa, S Nishio, K Suetsugu, M Hanks, A Stephens, Y Yokoi, K Nemoto, ***Hardness of braided quantum circuit optimization in the surface code***, IEEE Transactions on Quantum Engineering 4, 1-7

Optimizing DB circuit with ZX calculus

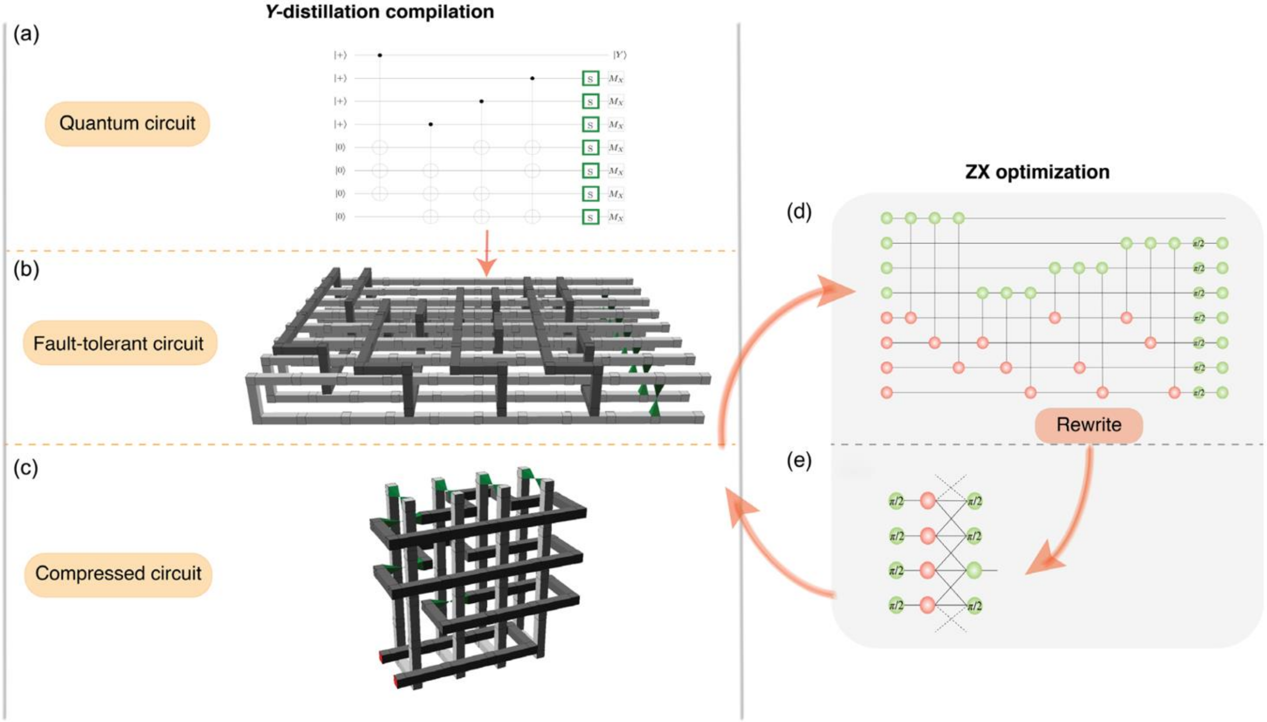


Fig. from [10]

[10] Michael Hanks, Marta Estarellas, William Munro & Kae Nemoto(2020). Effective compression of quantum braided circuits aided by ZX-calculus. *Physical Review X*, *10*(4), 041030.

2. Modular architecture

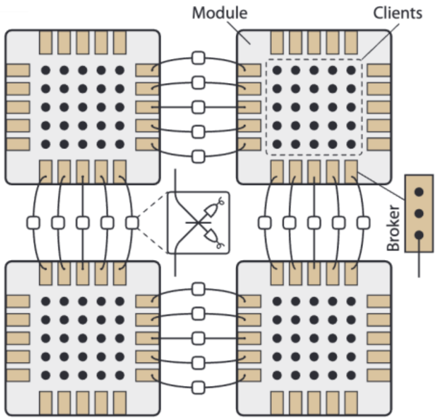


Fig. 1 in [12]

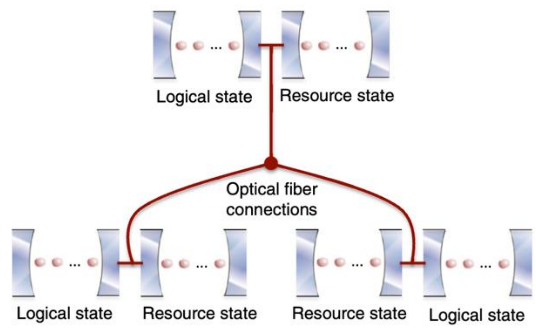
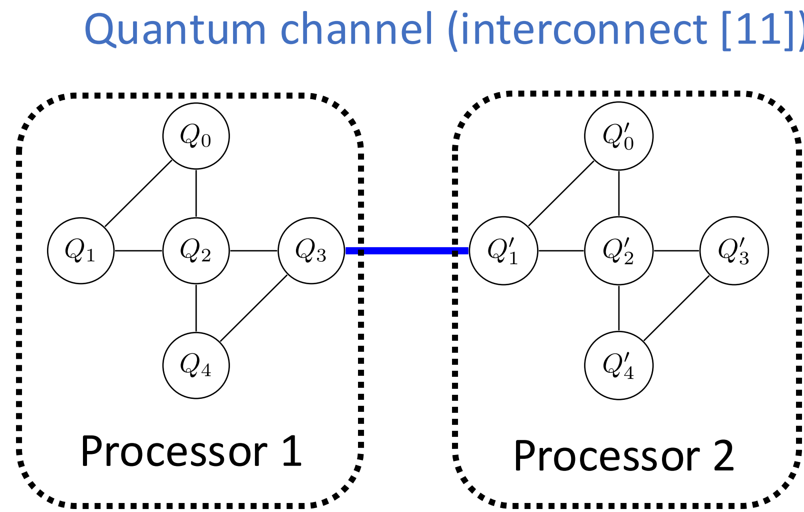


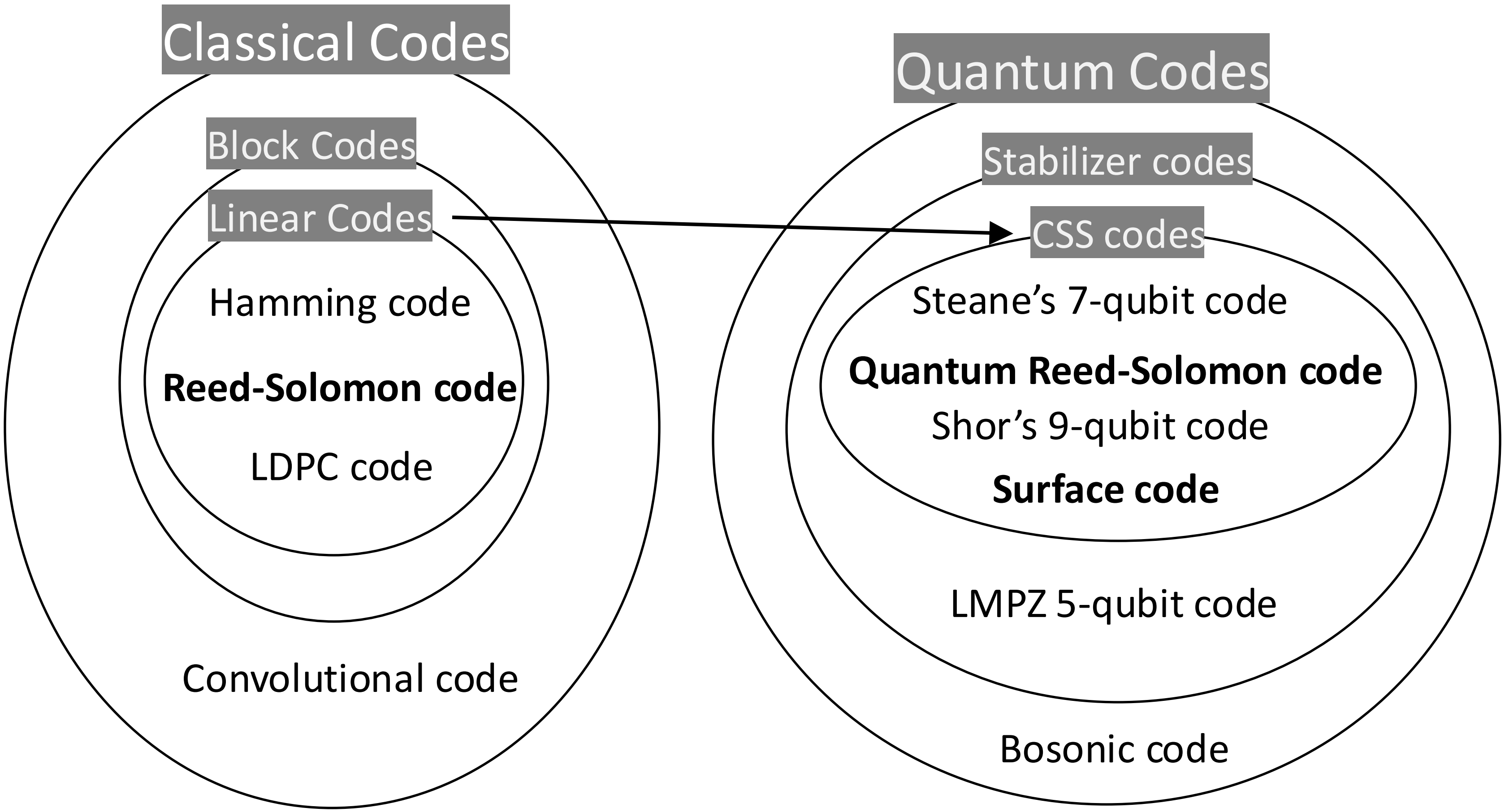
Fig. 3 in [13]



- photonic system is regarded as among the best candidate

[11] David Awschalom, et al. "Development of quantum interconnects (quics) for next-generation information technologies." *PRX Quantum* 2.1 (2021): 017002.
[12] Ying Li and Simon C. Benjamin. "Hierarchical surface code for network quantum computing with modules of arbitrary size." *Physical Review A* 94.4 (2016): 042303.
[13] Johannes Borregaard et al. "Efficient quantum computation in a network with probabilistic gates and logical encoding." *Physical Review A* 95.4 (2017): 042312.

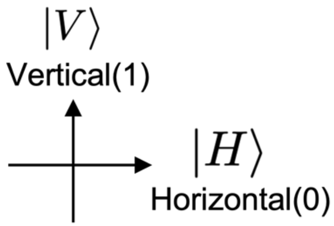
Quantum Error Correction Code

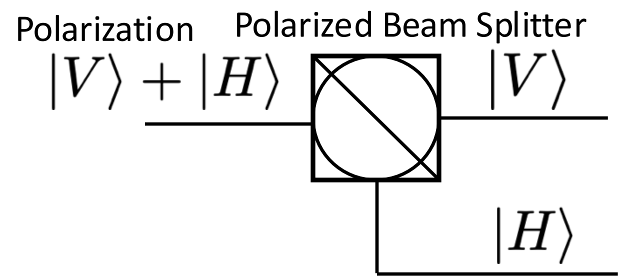


Optical Systems

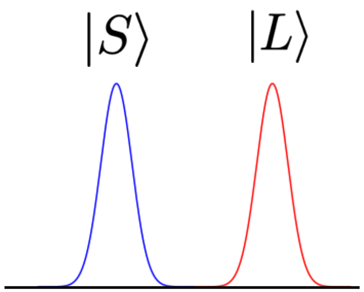
Treating the degrees of freedom of photons as qubits/qudits

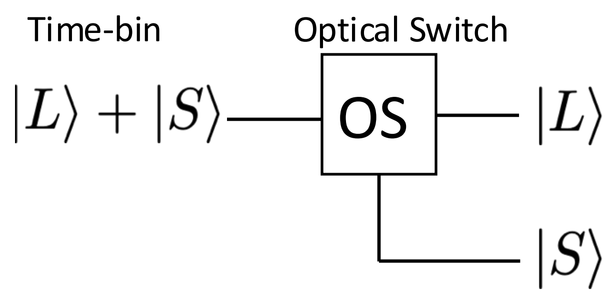
- Polarization





- Timebin





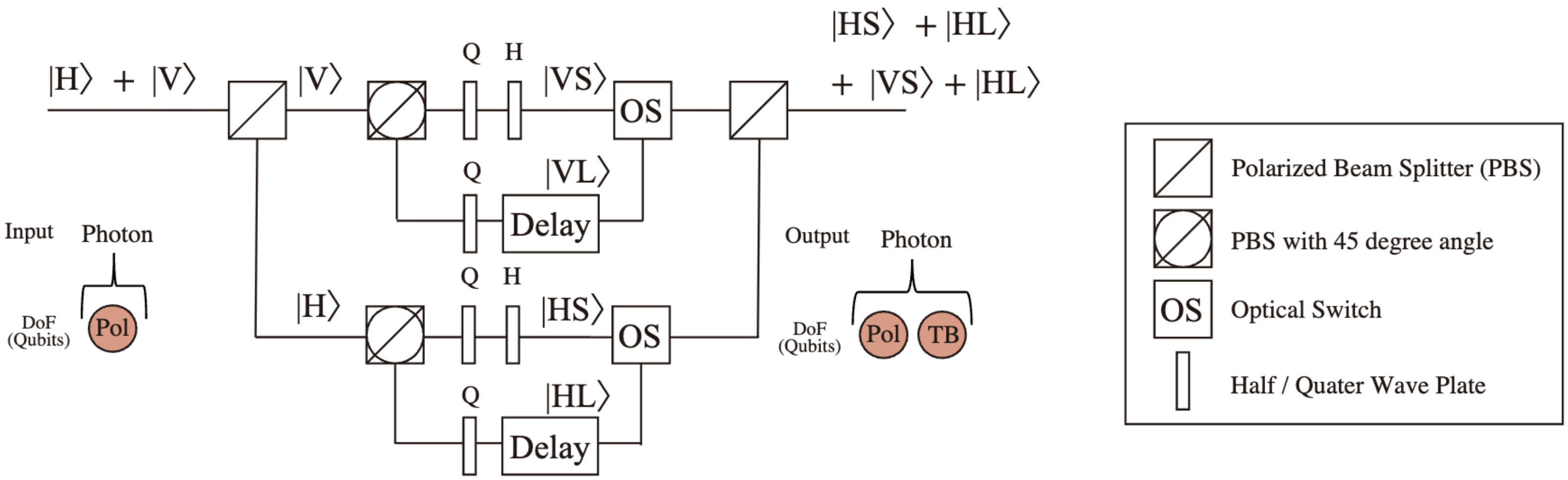
- Photon is sometimes lost, which is an obstacle to information processing(called loss error).
- Quantum Multiplexing[14]: By using multiple degree of freedom, one photon can be used as high-dimensional qudit.

e.g. polarization and timebin
$|VL\rangle$, $|VS\rangle$, $|HL\rangle$, $|VS\rangle$: 4-dim qudit

[14]Nicolò Lo Piparo, William J. Munro, and Kae Nemoto. "Quantum multiplexing." *Physical Review A* 99.2 (2019): 022337.

Timebin Encoding

- Timebin encoding



Quantum Multiplexing

Quantum Multiplexing is a technique for exploit multiple degrees of freedom of photon.

- It is known that QM can reduce the resource of QRS code
  - single photon sources[22]
  - # of CX gates for Toffoli gates[23]

[22] Nicolò Lo Piparo, et al. "Resource reduction for distributed quantum information processing using quantum multiplexed photons." *Physical Review Letters* 124.21 (2020): 210503.
[23] Nicolò Lo Piparo, et al. "Aggregating quantum networks." *Physical Review A* 102.5 (2020): 052613.

Quantum Multiplexing

Hao-Cheng Weng, Chih-Sung Chuu,
Implementation of Shor's Algorithm with a Single Photon in 32 Dimensions
arXiv:2408.08138

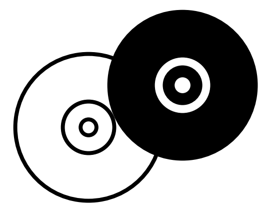
(Classical)Reed-Solomon code

- Errors are counted with respect to symbols (corresponds to the elements of Galois Field)

- Maximum distance separable code (satisfy singleton bound)

$$d_{min} = n - k + 1$$

- Be able to correct $t = n - k$ loss errors
  - good for optical channel



Applications

CD,DVD, QR, Satellite communication

Quantum Reed-Solomon code

- Good at correct loss errors for qudits
  - Application: quantum repeaters[15]

[15] Sreraman Muralidharan et al. "One-way quantum repeaters with quantum Reed-Solomon codes." *Physical Review A* 97.5 (2018): 052316.

Summary of Introduction

- QEC is a fundamental tool for Quantum Information Processing

- The implementation of quantum communication with optics is progressing.
  - there are still problems such as loss error.

- Quantum Reed-Solomon code is efficient for the loss errors
  - Errors are counted with respected to symbols(element of GF)

The cost of QEC including the encoding circuit makes their implementation quite challenging.

(Classical) Reed-Solomon Code

Reed-Solomon code over $GF(p^k)$ defined as follows:
where p is prime number and k is positive integer

1. Let α be the primitive element of $GF(p^k)$

$$GF(p^k) = \{0,1,\alpha,\alpha^2,\dots,\alpha^{p^k-2}\}$$

(Also define addition/multiplication over GF by using minimal polynomial)

2. Input information is treated as the coefficients of polynomial

$$a(x) = a_0 + a_1 x + \cdots a_{k-1}x^{k-1}$$

3. Generator polynomial is given as

$$g(x) = (x-\alpha^l)(x-\alpha^{l+1})\cdots(x-\alpha^{l+d-2})$$

where integer $l \geq 0$

4. Calculate codewords $w(x)$

$$w(x) = a(x)g(x)$$

Quantum Reed-Solomon code [18]

It is possible to construct Quantum Codes from Reed-Solomon code by using CSS construction.

e.g. construction from [19]

$$\mathcal{C}_1 = [d, k, d-k+1]_d$$

$$\mathcal{C}_2 = \mathcal{C}_1^{\perp} = [d, d-k, k+1]_d$$

$$\mathcal{Q}_{\mathcal{C}_1,\mathcal{C}_2} = \left[\left[\underbrace{d}_{\text{physical}}, \underbrace{2k-d}_{\text{logical}}, \underbrace{d-k+1}_{\text{distance}}\right]\right]_{\underset{\text{dim qudit}}{d}}$$

[18] Markus Grassl, Willi Geiselmann, and Thomas Beth. "Quantum reed-solomon codes." *International Symposium on Applied Algebra, Algebraic Algorithms, and Error-Correcting Codes*. Springer, Berlin, Heidelberg, 1999.

[19]Muralidharan Sreraman, et al. "One-way quantum repeaters with quantum Reed-Solomon codes." *Physical Review A* 97.5 (2018): 052316.

Phys. Rev. A **107**, 032620

Proposal: Encoding circuit (e.g. $[[5,1,3]]_5$ code)

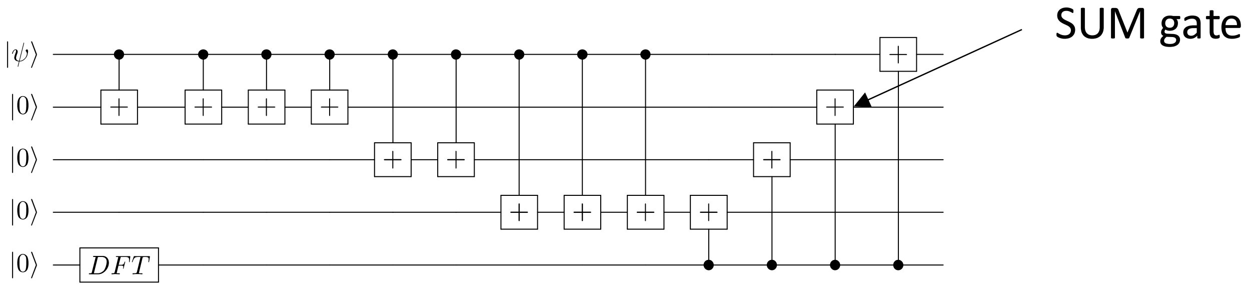
- new efficient implementation of the encoding circuit

$$\mathcal{C}_1 = [5,3,3]_5 \qquad \mathcal{C}_2 = [5,2,4]_5$$

$$G = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & \alpha & \alpha^2 & \alpha^3 & 1 \\ 0 & \alpha^2 & 1 & \alpha^2 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 2 & 4 & 3 & 1 \\ 0 & 4 & 1 & 4 & 1 \end{pmatrix} \quad H = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & \alpha & \alpha^2 & \alpha^3 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 2 & 4 & 3 & 1 \end{pmatrix}$$

$$X_L^1 = X_0^0 X_1^2 X_2^4 X_3^3 X_4^1$$

$$|0\rangle_L = |00000\rangle + |11111\rangle + |22222\rangle + |33333\rangle + |44444\rangle$$



5-dim qudit representation

Phys. Rev. A **107**, 032620

Operations for d-dimensional Qudit

Generalized CX gate[20]

$$SUM(|a\rangle\,|b\rangle) = |a\rangle\,|a + b\,mod\,d\rangle$$

Generalized Hadamard gate[21]

$$\mathrm{DFT}(|0\rangle) = |0\rangle + |1\rangle + \cdots + |\mathrm{d} - 1\rangle$$

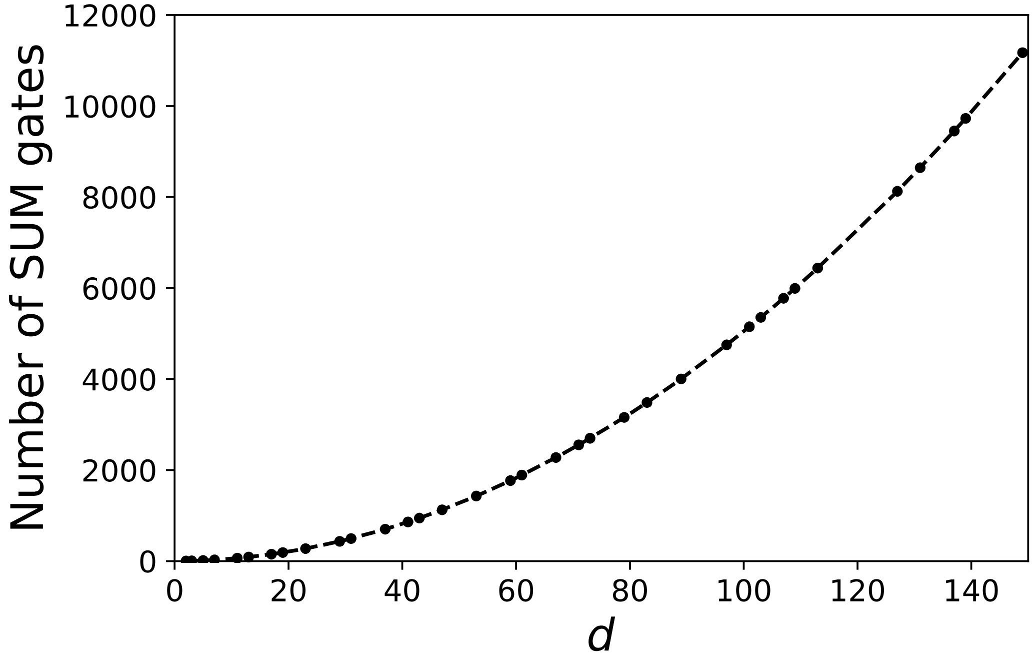
[20] Daniel Gottesman, Alexei Kitaev, and John Preskill. Encoding a qubit in an oscillator. Physical Review A,64(1):012310, 2001.
[21] Markus Grassl, Martin Rötteler, and Thomas Beth. "Efficient quantum circuits for non-qubit quantum error-correcting codes." *International Journal of Foundations of Computer Science* 14.05 (2003): 757-775.

Phys. Rev. A **107**, 032620

Result: Required resources

Required number of SUM gate for $\left[\left[d, 1, \left\lceil\frac{d+1}{2}\right\rceil\right]\right]_d$ Quantum Reed-Solomon Code over $GF(d)$

$$\frac{d^2+d-4}{2}$$



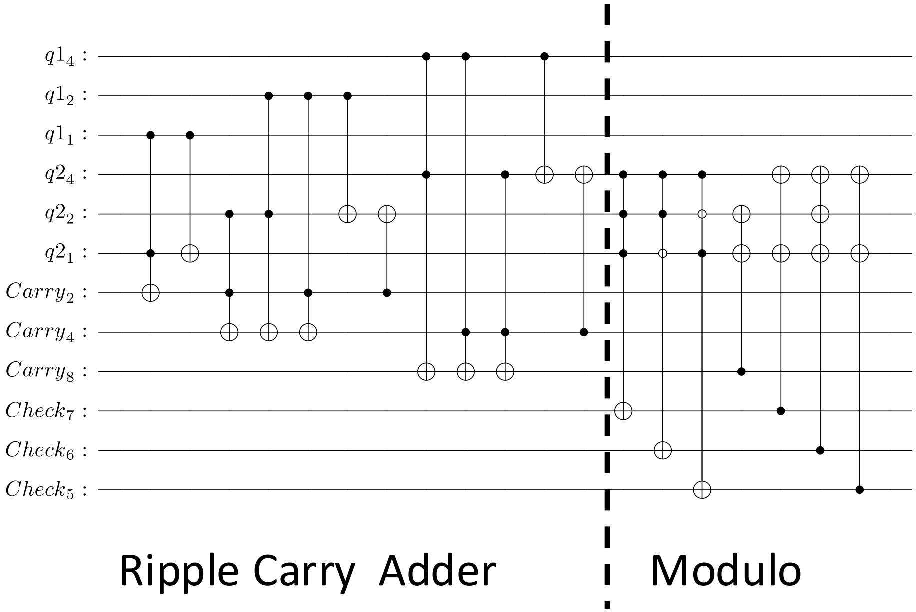
Phys. Rev. A **107**, 032620

Proposal: implementation of SUM Gate

Represent p-dim qudit over k-qubits system ($2^k \geq p$) where p is prime number

qudit	qubit
0	000
1	001
2	010
3	011
4	100

e.g. SUM gate for $p = 5$ (3-qubit mod 5 adder)



Numbers of Ancillae qubits

$k + 2(d-1)$ qubits

Number of gates

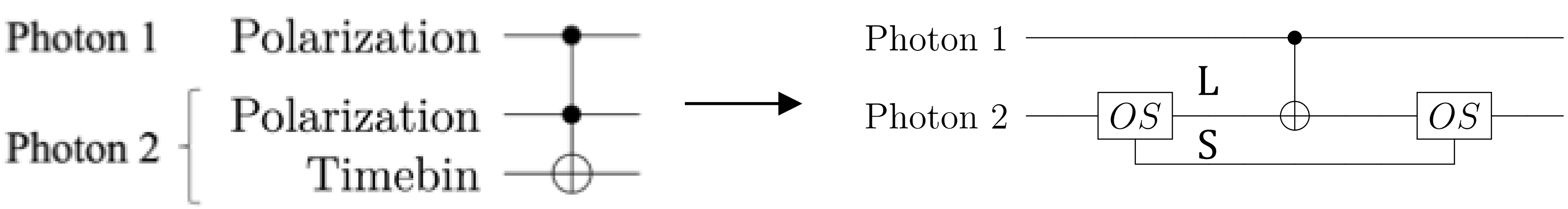
- RCA part

  $(CCX + CX) + (k-1)(3CCX + 2CX)$

- Modulo part

$$N_M = \begin{cases} \sum_{i=d}^{2(d-1)} \left(C_k X + H_D(\mathrm{bin}(i), \mathrm{bin}(i \bmod d))CX\right) & \text{if } 2(\ d-1) \leq 2^k \\ \sum_{d}^{2^k-1} C_k X + \sum_{2^k}^{2(d-1)} C_{k+1} X + \sum_{i=d}^{2(d-1)} H_D(\mathrm{bin}(i), \mathrm{bin}(i \bmod d))CX & \text{if } 2(\ d-1) > 2^k \end{cases}$$

Multiplexing reduction for the Toffoli gates [23]

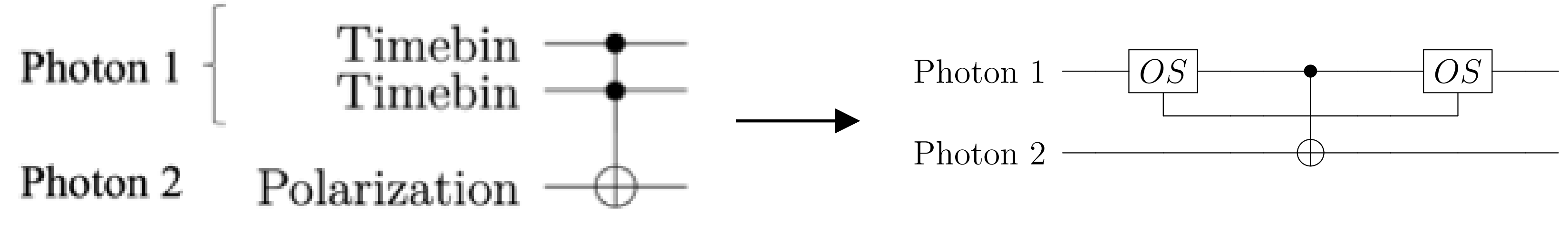


CCX gate
(Toffoli gate)

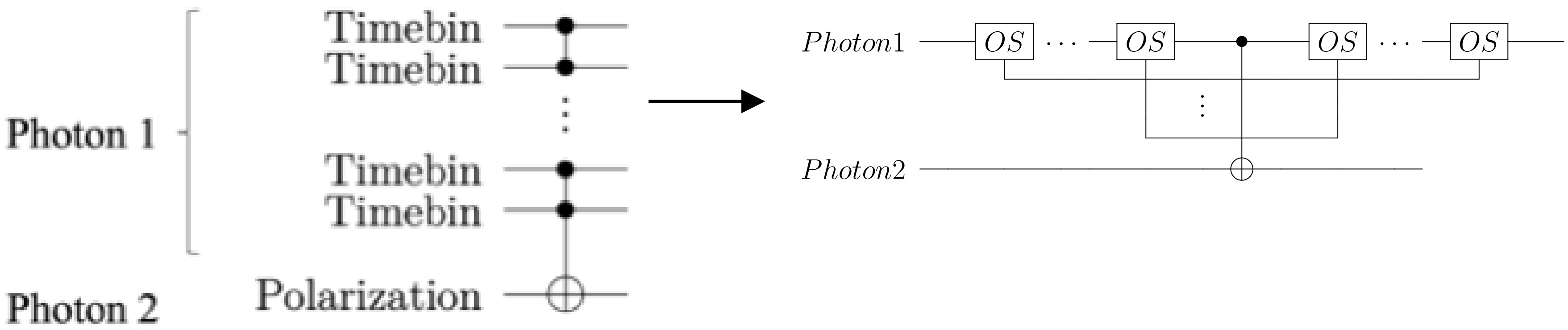
CX gate + OSs

[23] Nicolò Lo Piparo, et al. "Aggregating quantum networks." *Physical Review A* 102.5 (2020): 052613.

Proposal: Multiplexing decomposition for the C_kX gate



CCX



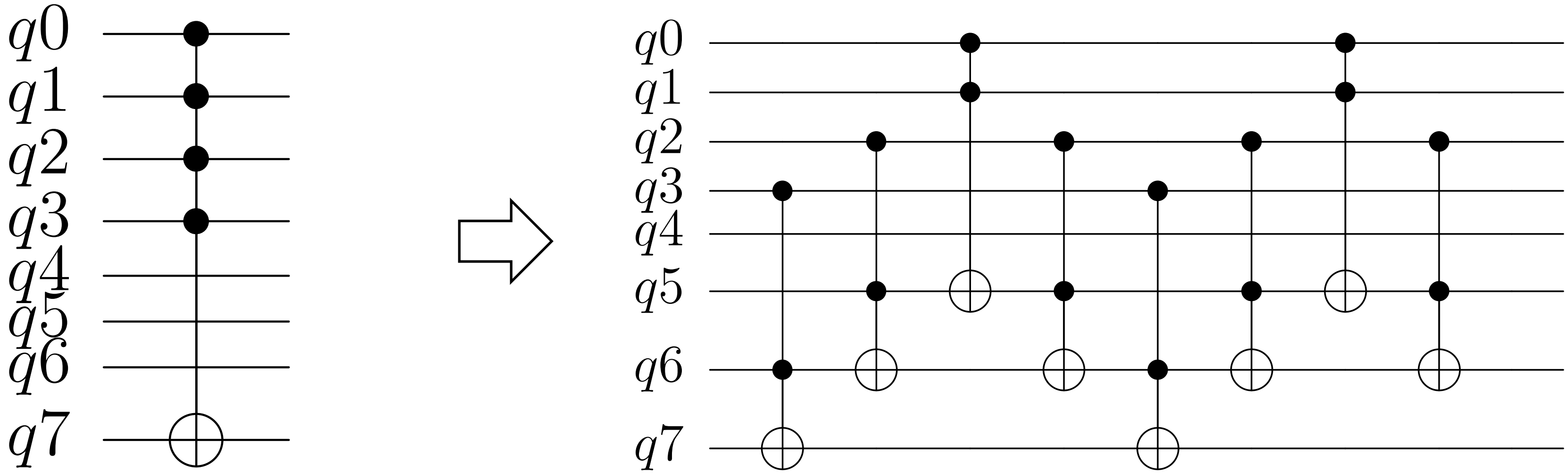
C_kX

Reduce # of control qubits by using OS

Phys. Rev. A **107**, 032620

Comparison: General Decomposition[24] with QM Decomposition

e.g.



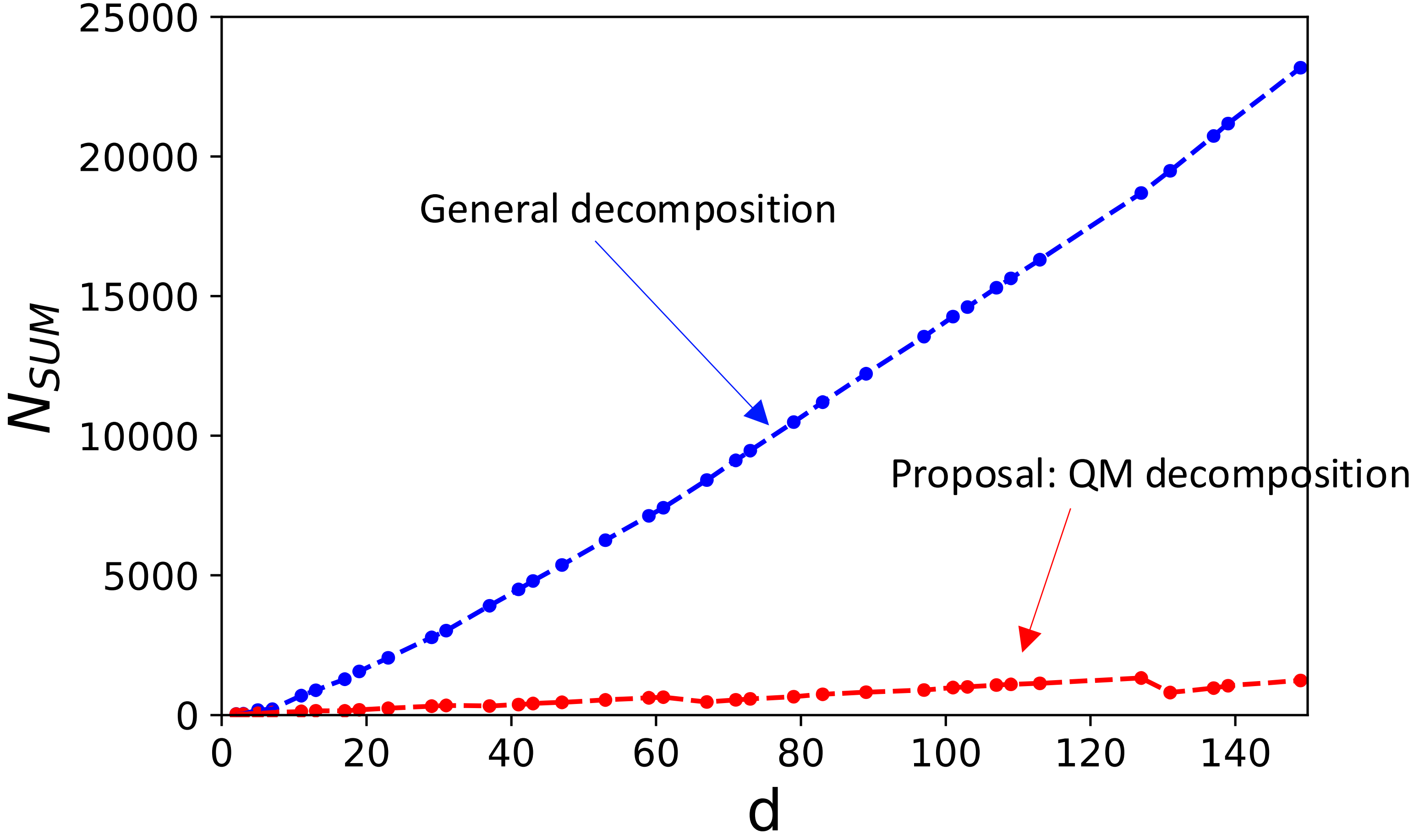
$$C_k X \rightarrow 4(k-2)CCX$$

$$CCX \rightarrow 6CX + 2H + 3T^{\dagger} + 5T$$

[24] A. Barenco, C. H. Bennett, R. Cleve, D. P. Di- Vincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter. Elementary gates for quantum computation. Physical Review A, 52(5):3457, 1995.

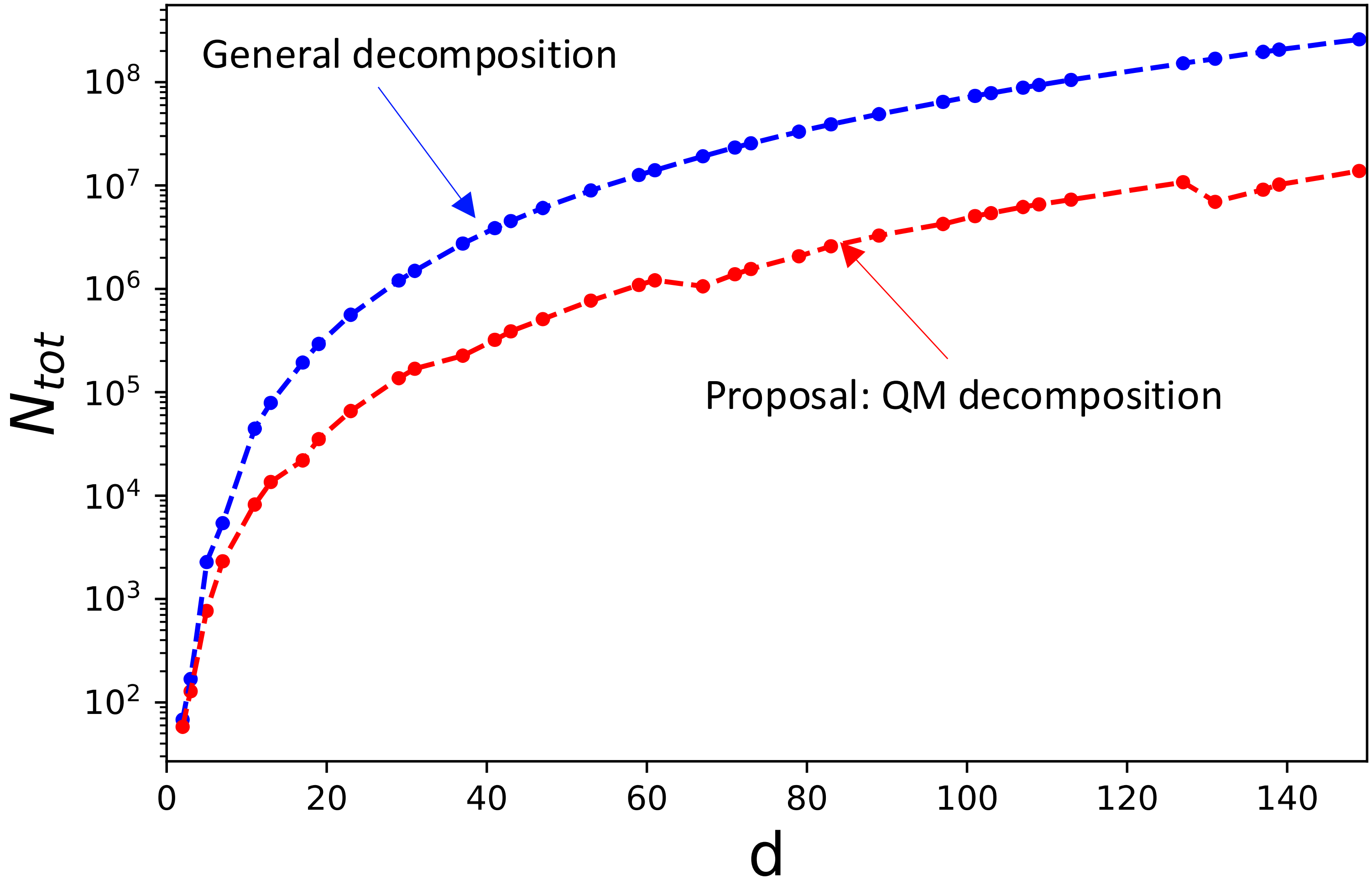
Phys. Rev. A **107**, 032620

Result: Multiplexing Optimization for SUM gates



Phys. Rev. A **107**, 032620

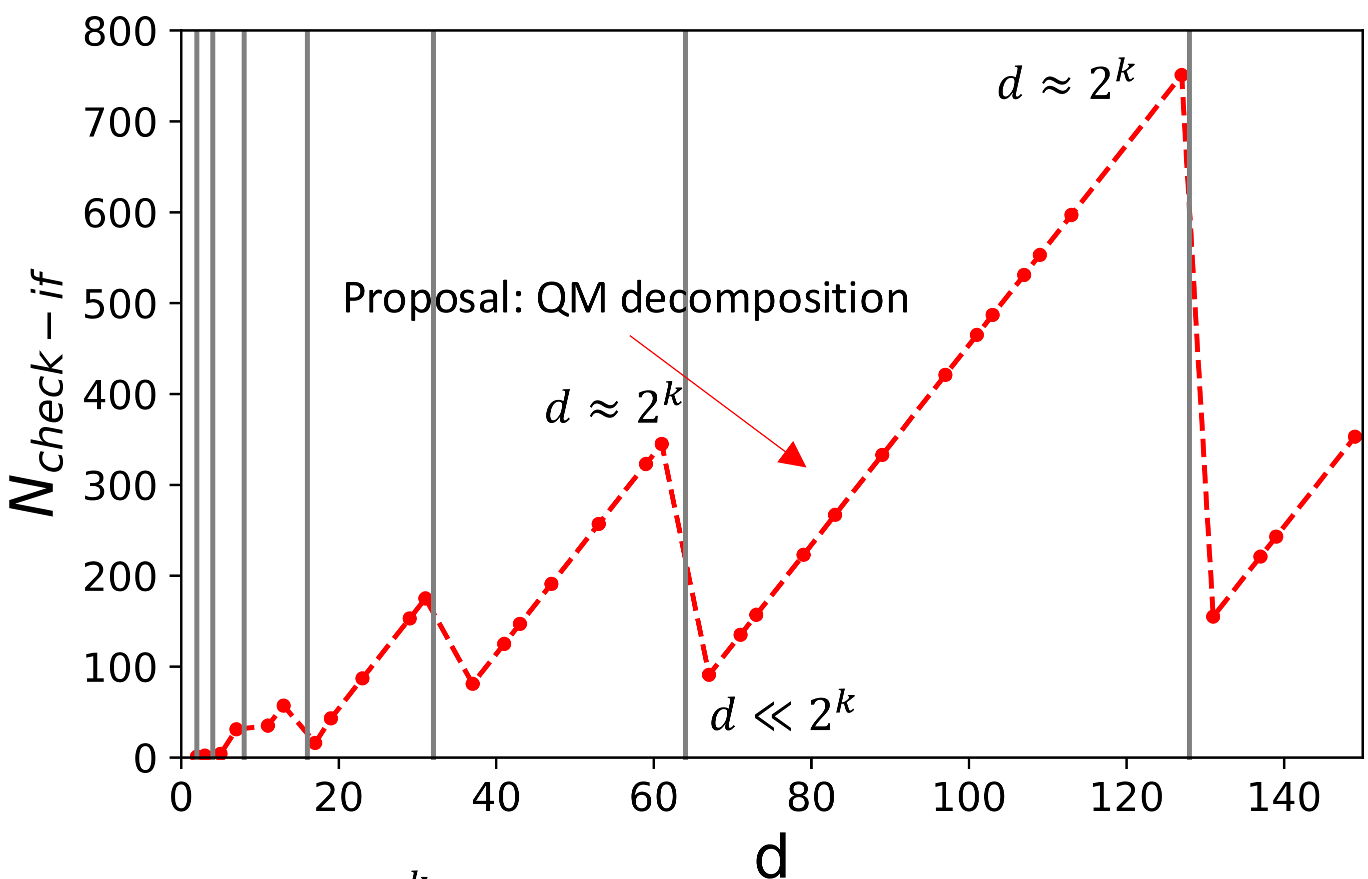
Result: Multiplexing Optimization for the Encoder



Result: The cost of SUM gate

Modulo part

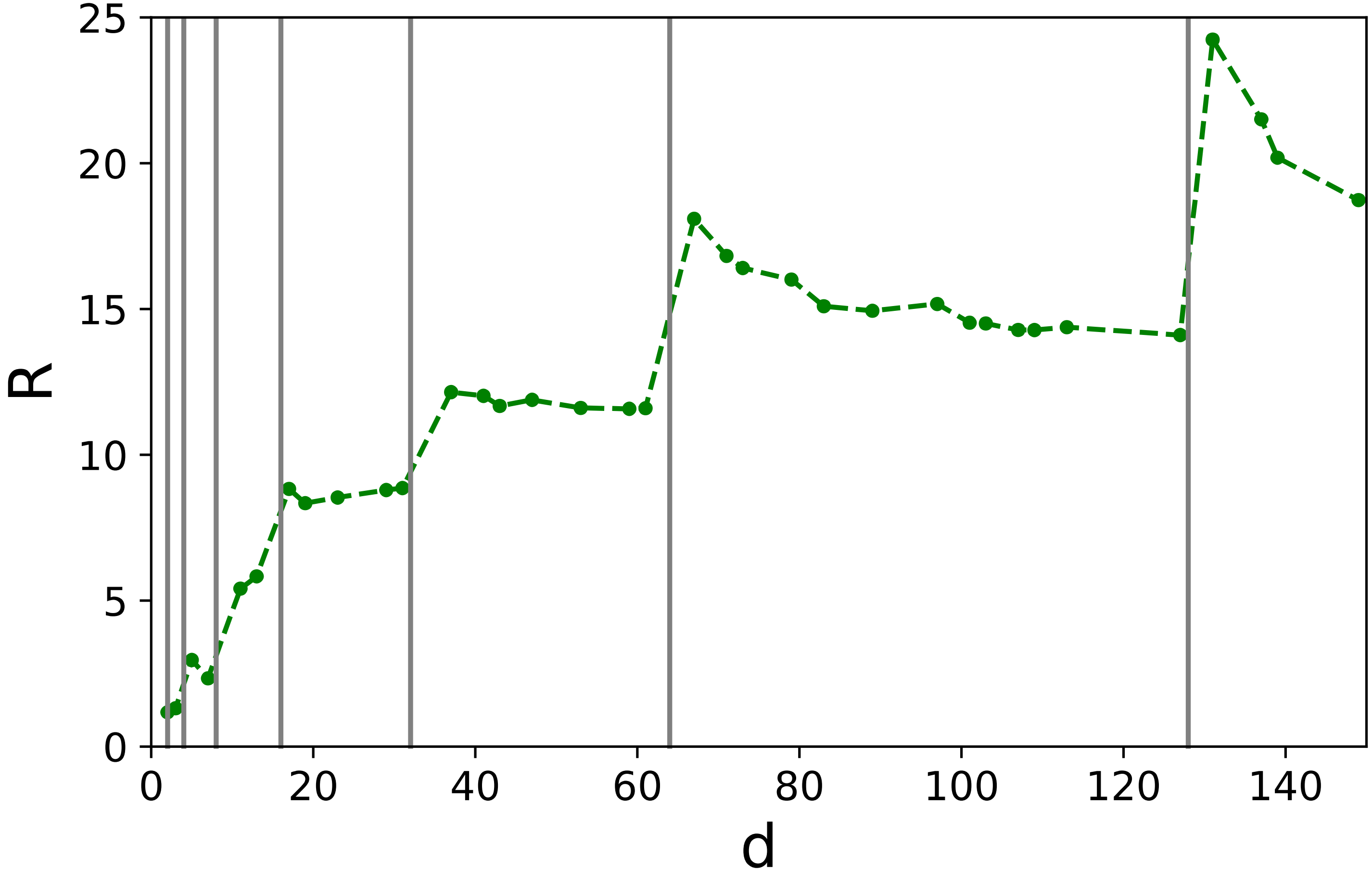
$$N_M = \begin{cases} \sum_{i=d}^{2(d-1)} \left(\underline{C_k X} + H_D(\mathrm{bin}(i), \mathrm{bin}(i \bmod d)) CX\right) & \text{if } 2(\mathrm{d}-1) \leqq 2^k \\ \sum_{d}^{2^k-1} \underline{C_k X} + \sum_{2^k}^{2(d-1)} \underline{C_{k+1} X} + \sum_{i=d}^{2(d-1)} H_D(\mathrm{bin}(i), \mathrm{bin}(i \bmod d)) CX & \text{if } 2(\mathrm{d}-1) > 2^k \end{cases}$$



Gray line ：d = 2^k

Phys. Rev. A **107**, 032620

Result: Multiplexing Optimization Ratio



Gray line ： dim = 2^k

Phys. Rev. A **107**, 032620

Conclusion for QM for QRS

- We apply QM to the implementation of the encoding circuit of the QRS codes
- In this case we show that $C_k X$ gates can be reduced into a single CX gate by using linear optical elements.
- Therefore, the total number of CX gates require to implement d-dimensional QRS codes is drastically reduced (except when $d = 2^m$).
- We believe that this can also be applied to other systems such as Grover's search and discrete time quantum walks, leading to a much more feasible implementation of such technology

Phys. Rev. A **107**, 032620

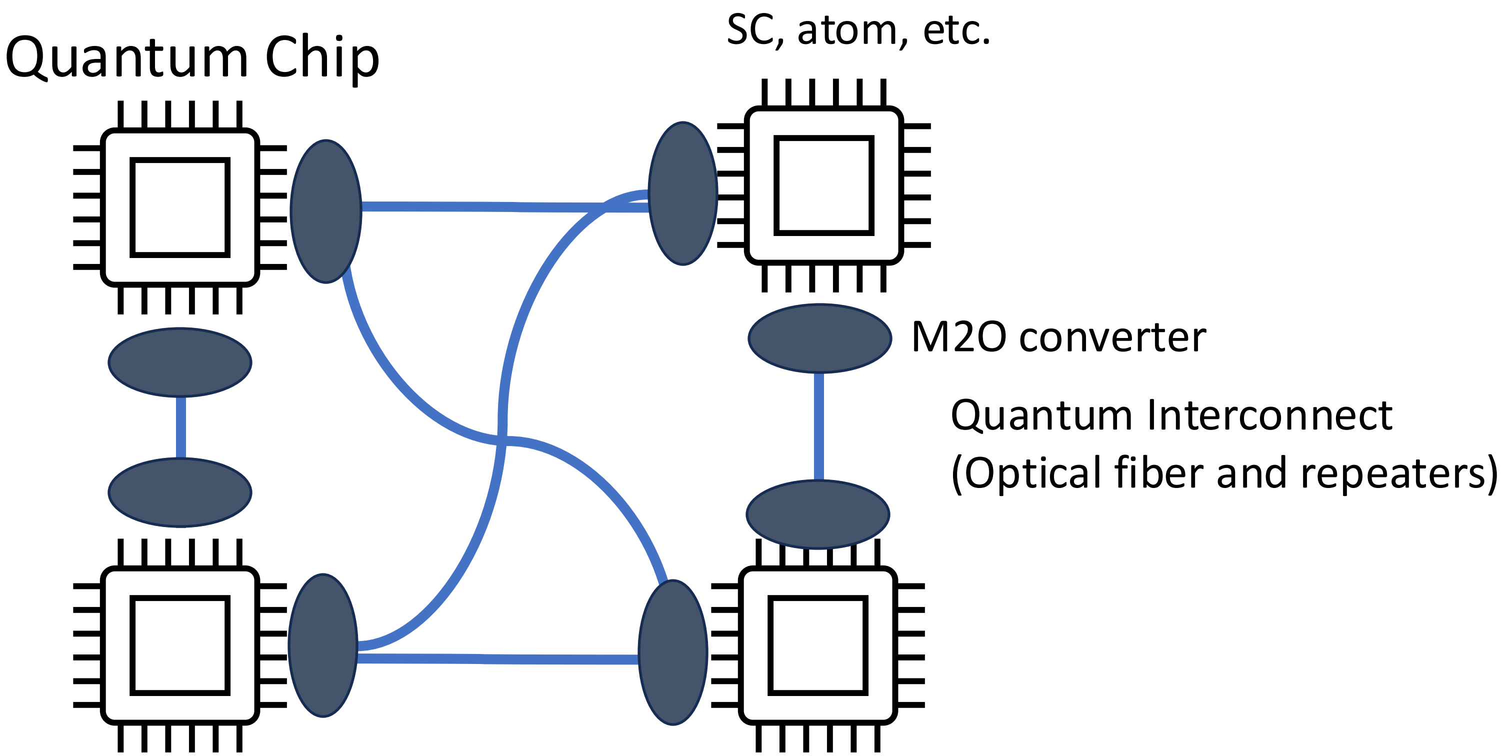
Future work

- Evaluation of other resources (OSs)
- Correction/Decoder circuit
- Comparison with other circuit optimization methods

Applications

- Other CSS codes and more general codes
- Any system that requires SUM gates

QM and surface codes



	Quantum Chip	Quantum Interconnect
Case 1	Topological Codes	MDS codes(e.g. QRS)
Case 2	**Topological Codes**	

Hybrid codes system may introduce overhead...

Erasure channel

Erasure(photon loss) is dominant in the optical systems

$$\rho \to (1-\varepsilon)\rho + \underbrace{\varepsilon|e\rangle\langle e|}_{\text{Erasure error}}$$

where $|e\rangle \notin \mathcal{H}_2$

→We can detect erasure without destruction!

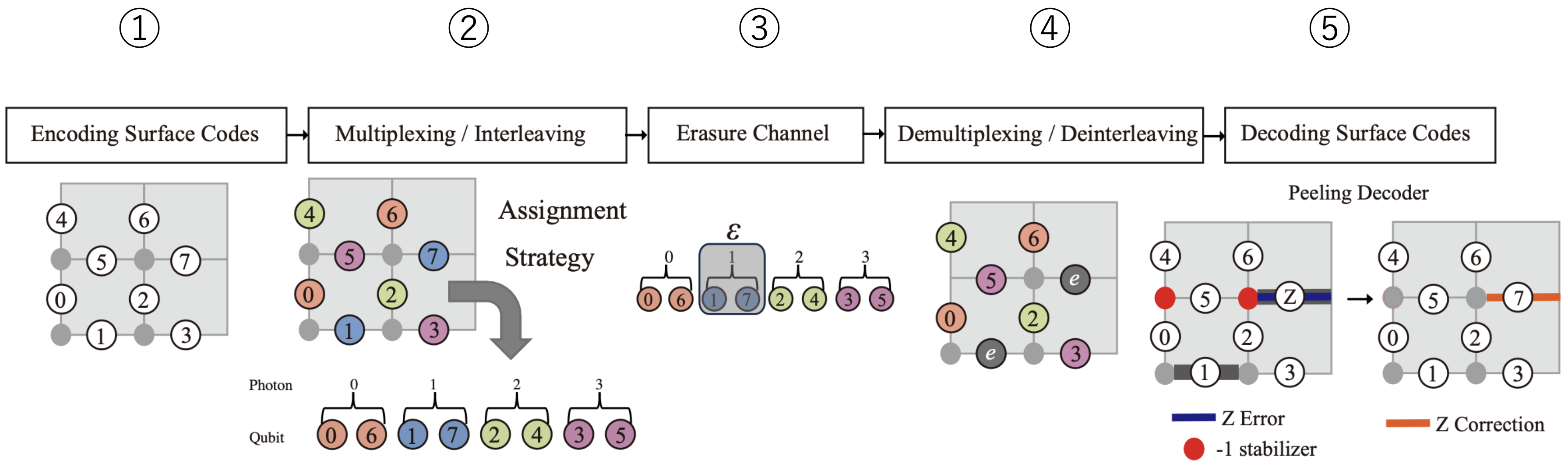
0. detect erasure errors
1. replace the erased qubit with a mixed-state

$$\frac{\mathbb{I}}{2} = \frac{1}{4}(\rho + X\rho X + Y\rho Y + Z\rho Z)$$

2. stabilizer measurement
3. Decoding (calculate syndrome and correct)

arXiv:**2406.08832**

Proposal Quantum multiplexed SC comm



Erasure channel

- Erasure(photon loss) is dominant in the optical systems

$$\rho \rightarrow (1-\varepsilon)\rho + \underbrace{\varepsilon|e\rangle\langle e|}_{\text{Erasure error}}$$

where $|e\rangle \notin \mathcal{H}_2$

→We can detect erasure without destruction!

Correction procedure for erasure errors

$$\rho \to (1-\varepsilon)\rho + \varepsilon|e\rangle\langle e|$$

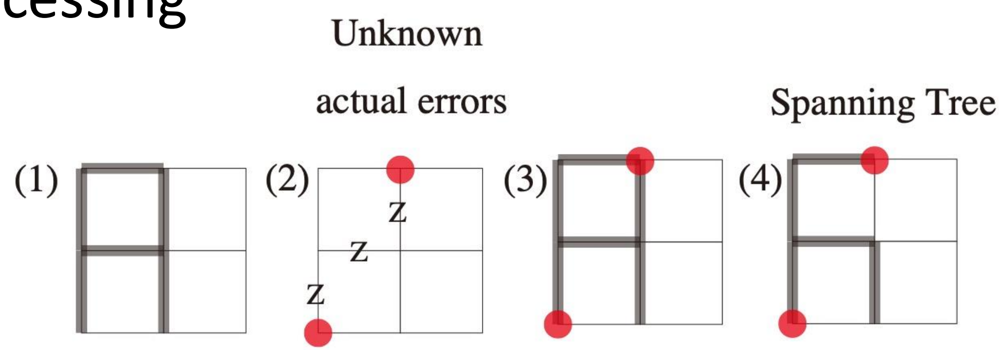
0. detect erasure errors
1. replace the erased qubit with a mixed-state

$$\frac{\mathbb{I}}{2} = \frac{1}{4}(\rho + X\rho X + Y\rho Y + Z\rho Z)$$

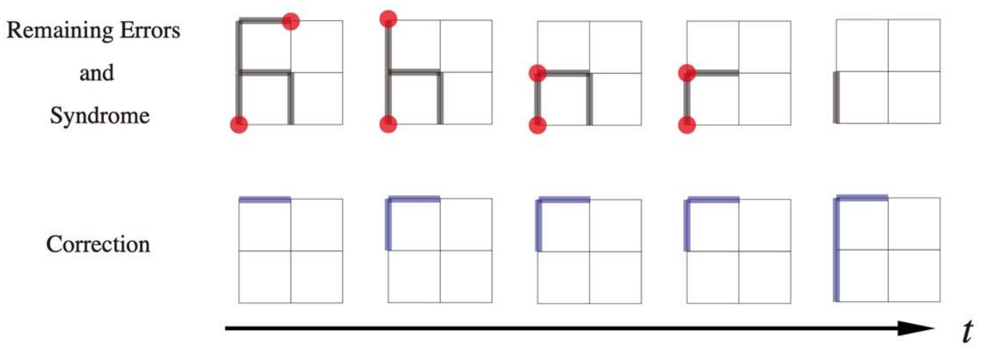
2. stabilizer measurement
3. Decoding (calculate syndrome and correct)

Erasure Correction with Surface Codes

Preprocessing



Peeling decoder [25]

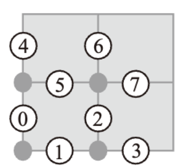
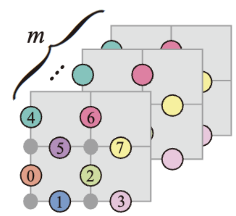
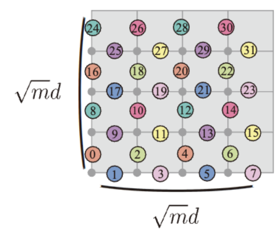
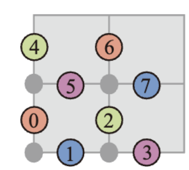


[25] N. Delfosse and G. Zémor, Linear-time maximum likelihood decoding of surface codes over the quantum erasure channel, Physical Review Research 2, 033042 (2020)

Proposal: Three scenarios for QM Comm

Suppose we apply quantum multiplexing with m qubits per one photon, we can send

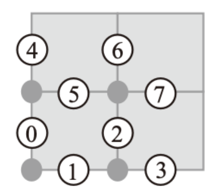
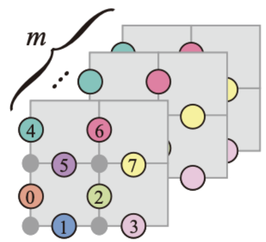
1. m different code words with the same # of photons
2. $\sqrt{m} \times \sqrt{m}$ bigger code words with the same # of photons
3. Original code words with $\frac{1}{m}$ photons

Scenarios	 without multiplexing	 (A)	 (B)	 (C)
Code parameters	$[[2d^2, 1, d]]$	$[[2d^2, 1, d]]$	$[[2md^2, 1, \sqrt{md}]]$	$[[2d^2, 1, d]]$
Number of Codes	1	m	1	1
Number of Data Qubits	$2d^2$	$2md^2$	$2md^2$	$2d^2$
Number of Photons	$2d^2$	$2d^2$	$2d^2$	$\lfloor 2d^2/m \rfloor$
Logical Error Rate	-	Same as without quantum multiplexing	Affected by classical correlation	Affected by classical correlation

Scenario (A) Multiple Code Words

Sending m code words

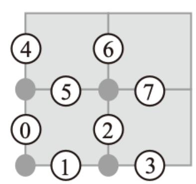
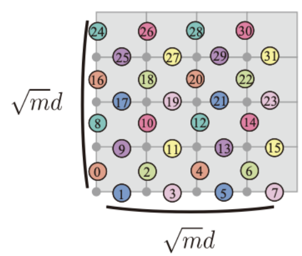
- m times better throughput
- No classical correlation inside a code

Scenarios	 without multiplexing	 (A)
Code parameters	$[[2d^2, 1, d]]$	$[[2d^2, 1, d]]$
Number of Codes	1	m
Number of Data Qubits	$2d^2$	$2md^2$
Number of Photons	$2d^2$	$2d^2$
Logical Error Rate	-	Same as without quantum multiplexing

Scenario (B) Large Code Word

Sending bigger codes
- Large code distance
- Encoding m qubits in a photon

→ correlated Pauli errors

Scenarios	 without multiplexing	 (B)
Code parameters	$[[2d^2, 1, d]]$	$[[2md^2, 1, \sqrt{md}]]$
Number of Codes	1	1
Number of Data Qubits	$2d^2$	$2md^2$
Number of Photons	$2d^2$	$2d^2$
Logical Error Rate	-	Affected by classical correlation

Scenario (B) Large Code Word

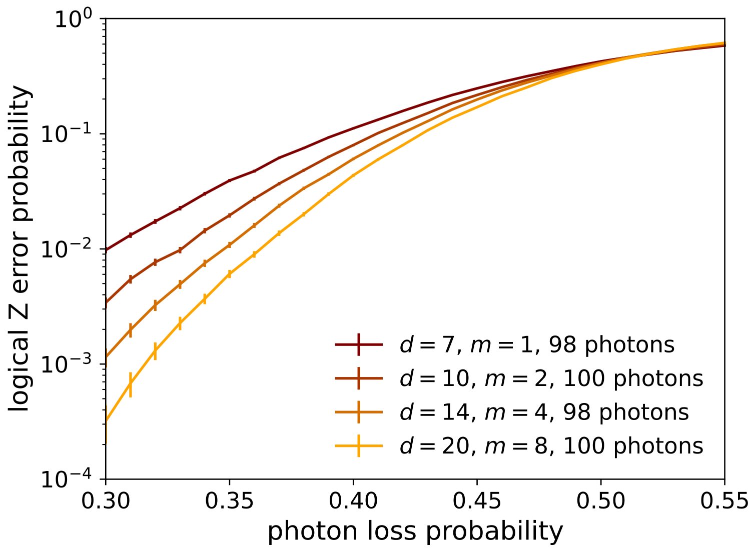
arXiv:**2406.08832**

Sending bigger codes

- Large code distance
- Encoding m qubits in a photon

→ correlated Pauli errors

Benefit from the large distance is dominant



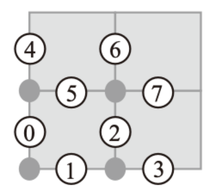
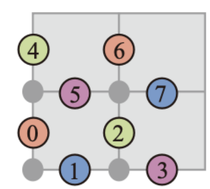
Scenario (C)

arXiv:**2406.08832**

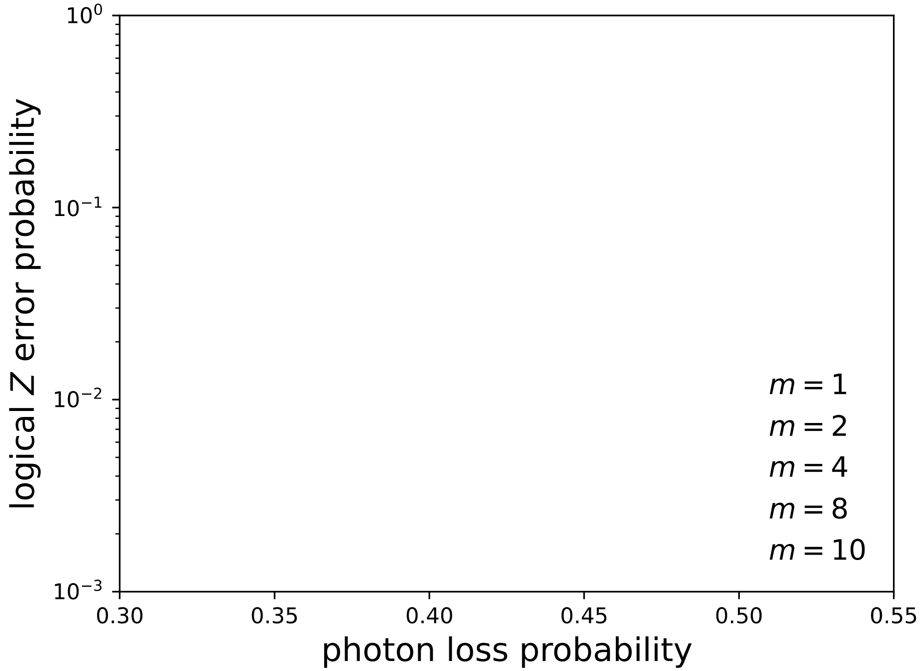
Sending same codes with fewer photons

- m times better throughput
- Encoding m qubits in a photon

→ Classical correlation inside codewords

Scenarios	 without multiplexing	 (C)
Code parameters	$[[2d^2, 1, d]]$	$[[2d^2, 1, d]]$
Number of Codes	1	1
Number of Data Qubits	$2d^2$	$2d^2$
Number of Photons	$2d^2$	$\lfloor 2d^2/m \rfloor$
Logical Error Rate	-	Affected by classical correlation

Scenario (C): Degradation caused by QM



- There is a degree of freedom in the assignment of qubits to photons

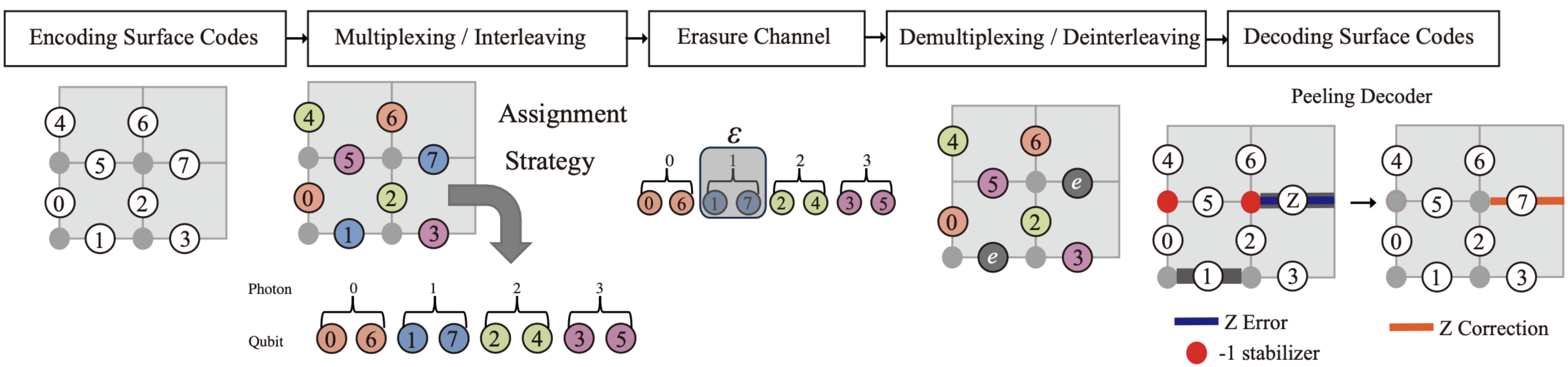
→ We propose 5 assignment strategies

Assignment strategy

arXiv:**2406.08832**

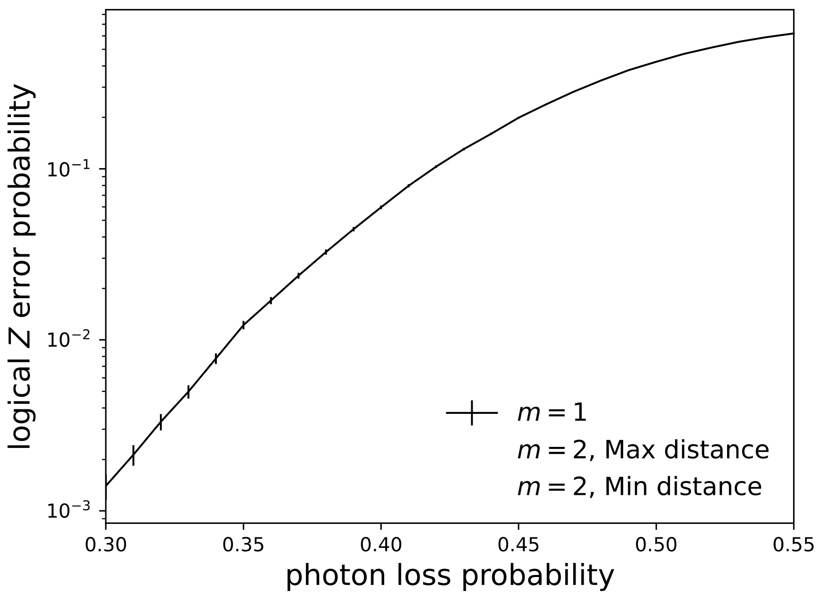
The impact of introducing correlation on the performance is not trivial.

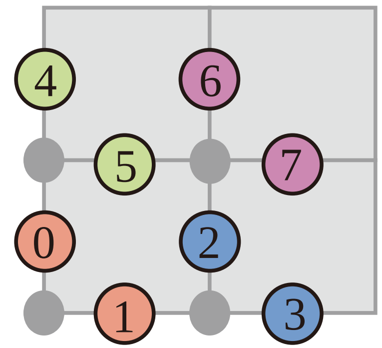
- Can negative effects of correlation be avoided?
- Is it possible to achieve better performance than no-multiplexing with reducing number of photons?



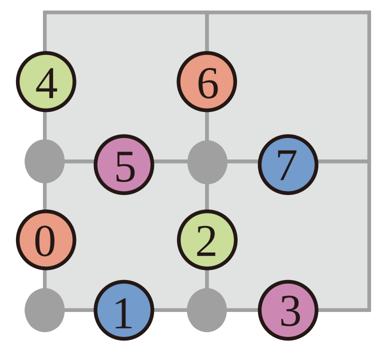
Assignment strategies for QM

arXiv:**2406.08832**





(i) Minimum distance



(ii) Maximum distance

Maximizing distance may help!

Assignment strategies for QM (2)

arXiv:**2406.08832**

(iii) Uniformly random

Inspired by interleaving for classical error correction codes

(iv) Random + Thresholds

For photons:

Threshold

$$T = \frac{d}{2} - 1$$

Pick 1st qubit randomly

While # of qubits in the photon < m:

Pick a candidate qubit c randomly.

if c has distance $> T$ from nearest member:

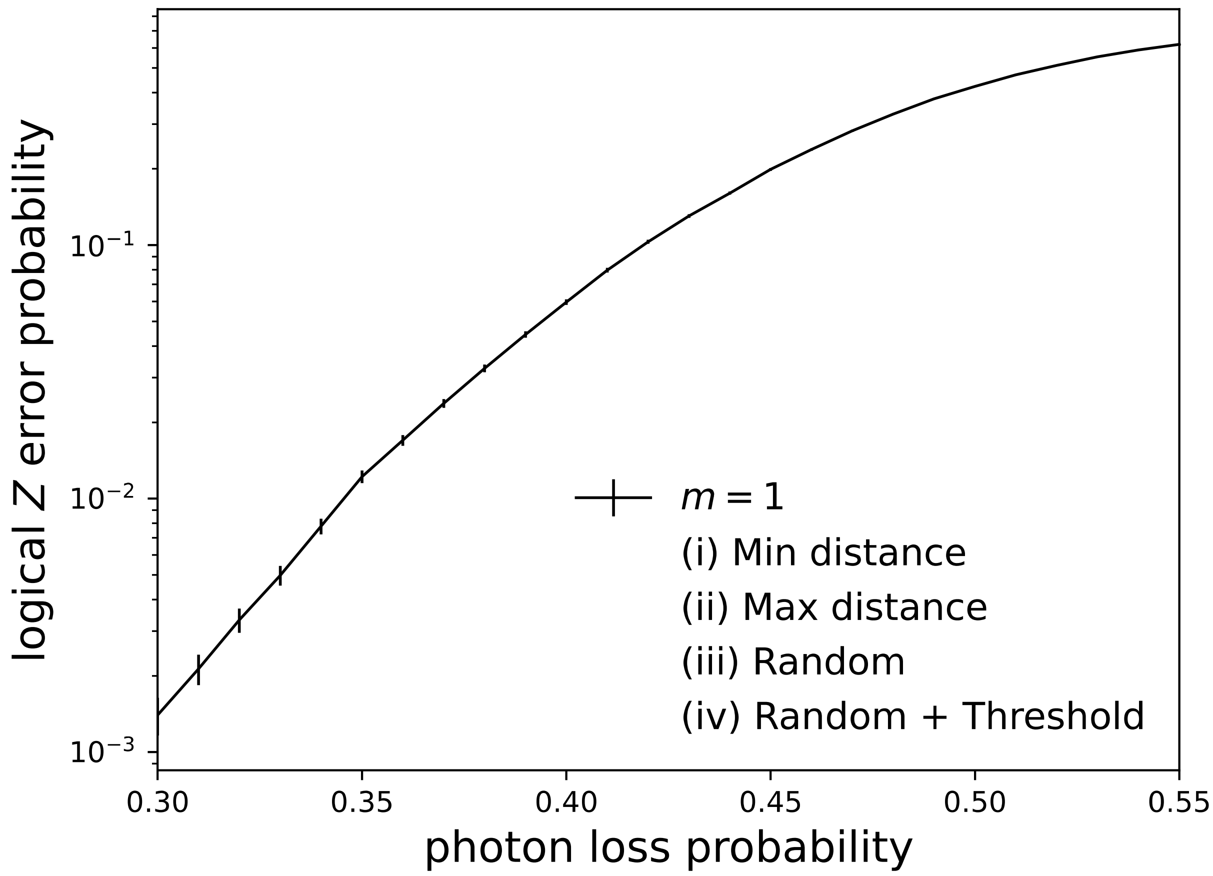
add c to the photon

if there is no candidate:

update T as $T - 1$

Comparison on assignment strategies

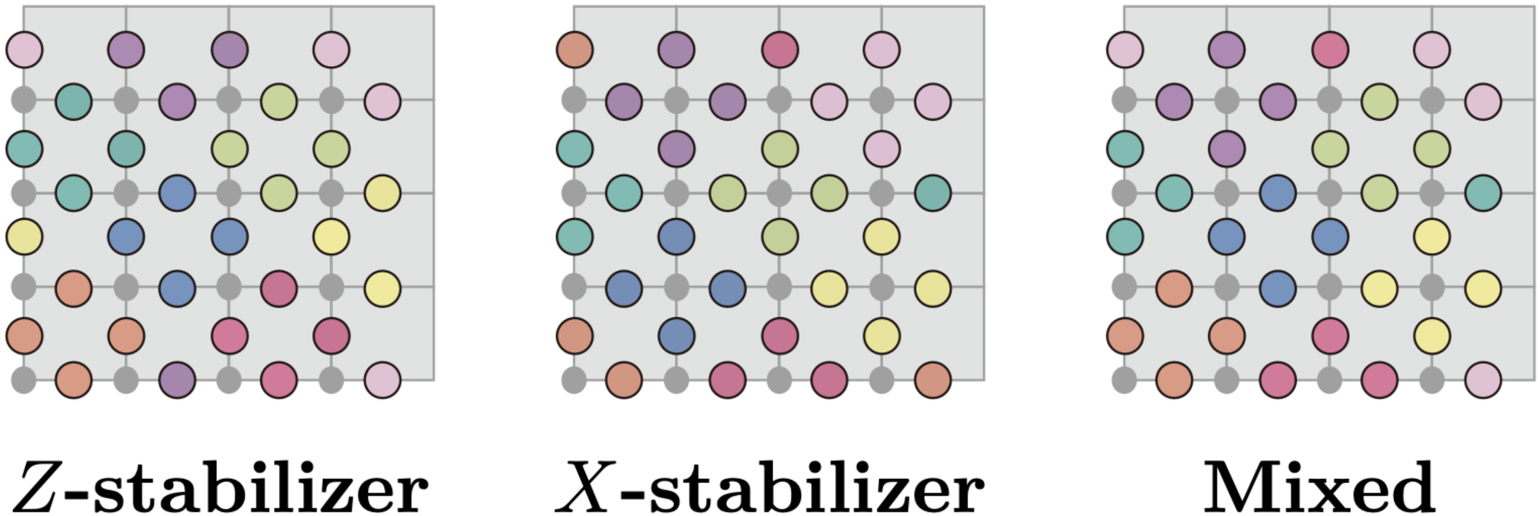
$m = 2$



- Randomness and maximizing the distance has positive effect

Assignment strategies for QM

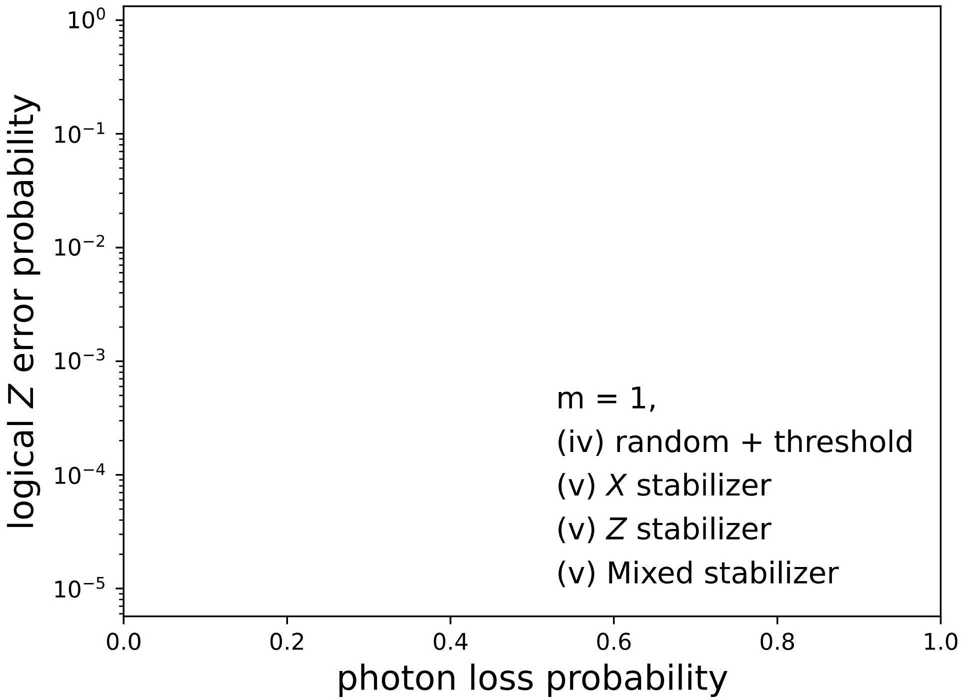
(v) Stabilizer based



Assignment strategies for QM

arXiv:2406.08832

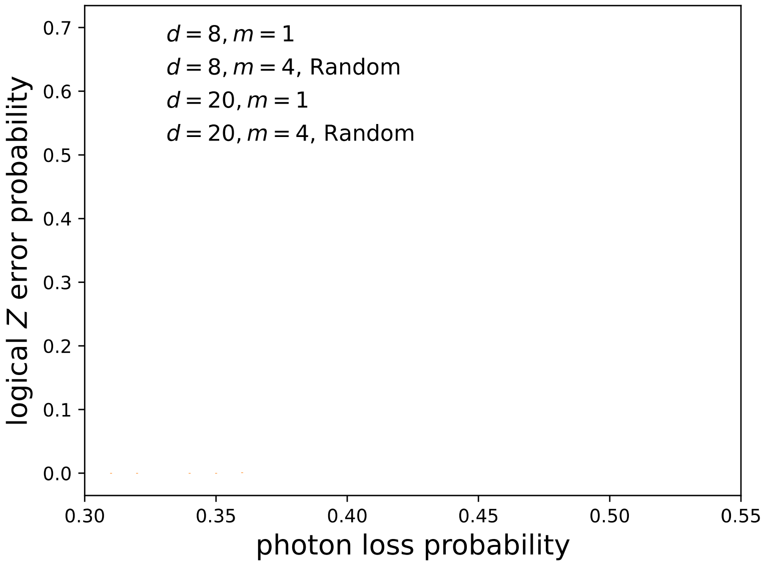
$m = 4$

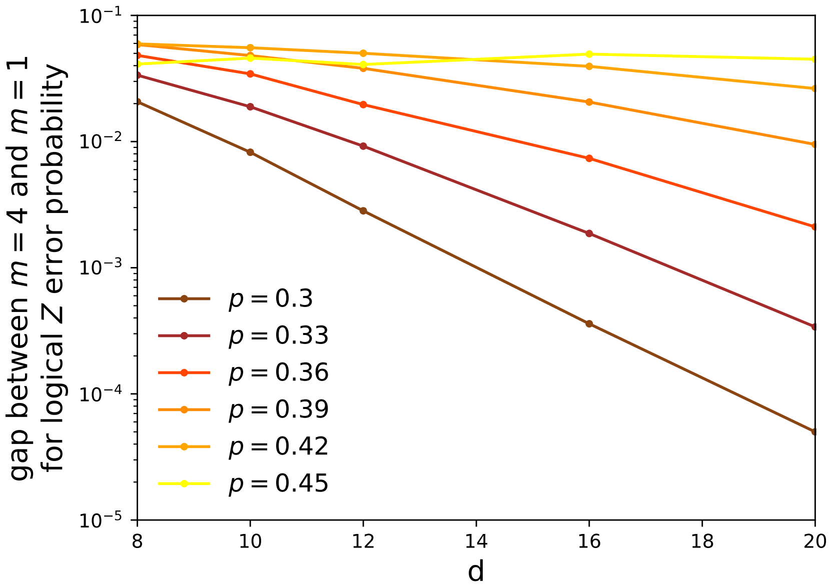


- May be useful when using biased code or error is biased

Scaling the code

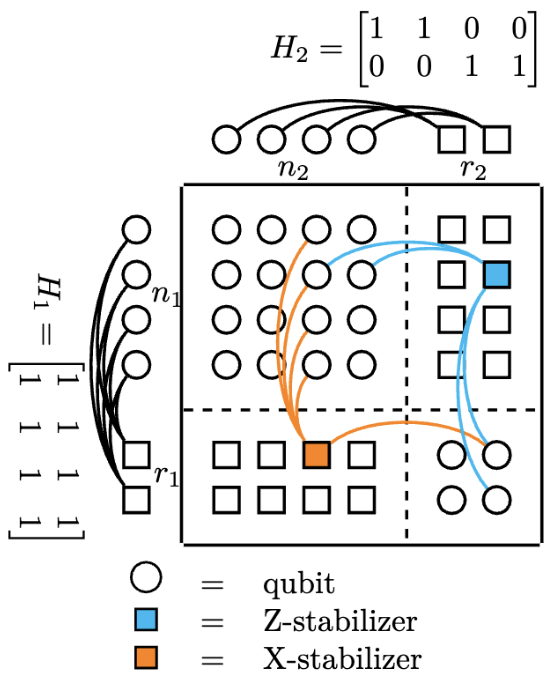
If $m \ll d$, classical correlation can be ignored



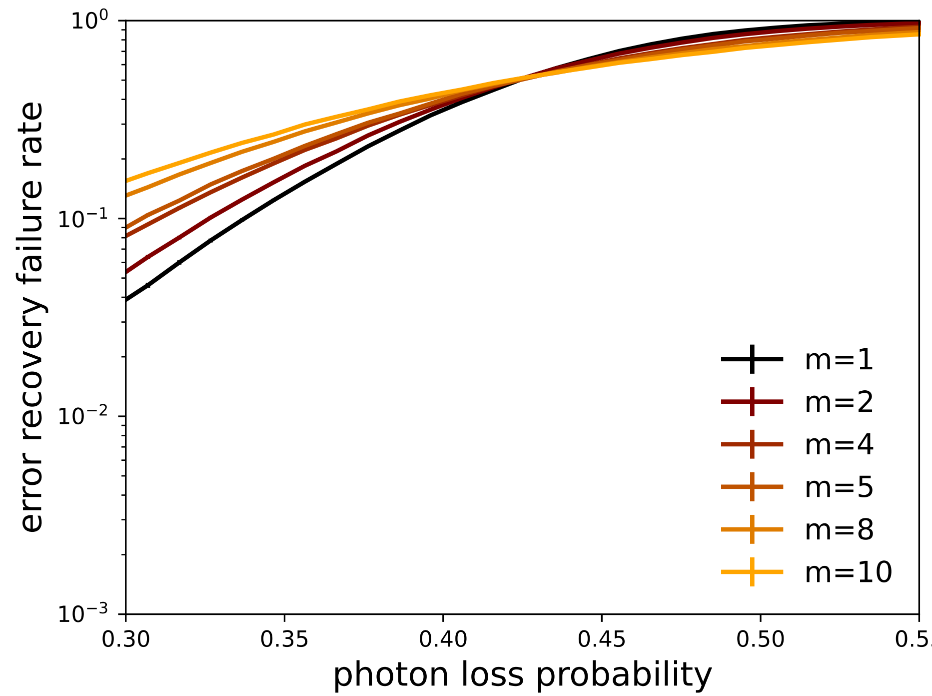


Hypergraph Product (HGP) Codes

- Asymptotically finite rate
- Can be regarded as Generalized surface code
  Product of two ring graphs



With random assignment



$$H_X = \left[\begin{array}{cc} 1000100010001000 & 1000 \\ 0100010001000100 & 1000 \\ 0010001000100010 & 0100 \\ 0001000100010001 & 0100 \\ 1000100010001000 & 0010 \\ 0100010001000100 & 0010 \\ 0010001000100010 & 0001 \\ 0001000100010001 & 0001 \end{array}\right] \quad H_Z = \left[\begin{array}{cc} 1100000000000000 & 1010 \\ 0011000000000000 & 0101 \\ 0000110000000000 & 1010 \\ 0000001100000000 & 0101 \\ 0000000011000000 & 1010 \\ 0000000000110000 & 0101 \\ 0000000000001100 & 1010 \\ 0000000000000011 & 0101 \end{array}\right]$$

Non-ML Decoder

Maximally-likelihood Decoder for HGP (Gaussian elimination)

$O(n^3)$

Combined decoder (peeling + pruned peeling + VH)[26]

$O(n^2)$ or $O(n^{1.5})$ for probabilistic version

Fig from [26]

[26]N. Connolly, V. Londe, A. Leverrier, and N. Delfosse, “Fast erasure decoder for hypergraph product codes,” Quantum, vol. 8, p. 1450, Aug. 2024, doi: 10.22331/q-2024-08-27-1450.

Decoding stopping set

Combined decoder[26] runs in $O(n^2)$

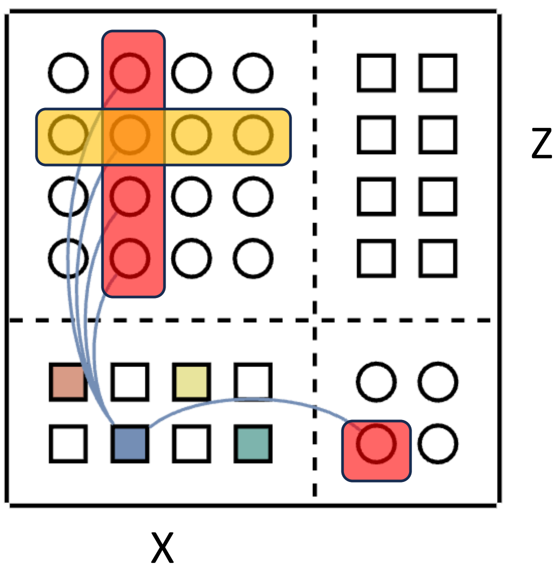
Two types of stopping set

- Stabilizer stopping set

  All qubits in support of the x/z stabilizer

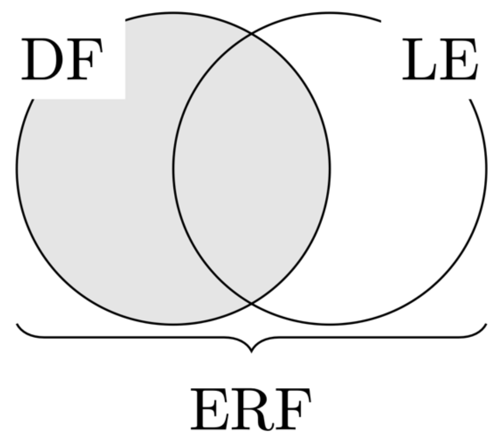
- Classical stopping set

  entire low or column: classical codes



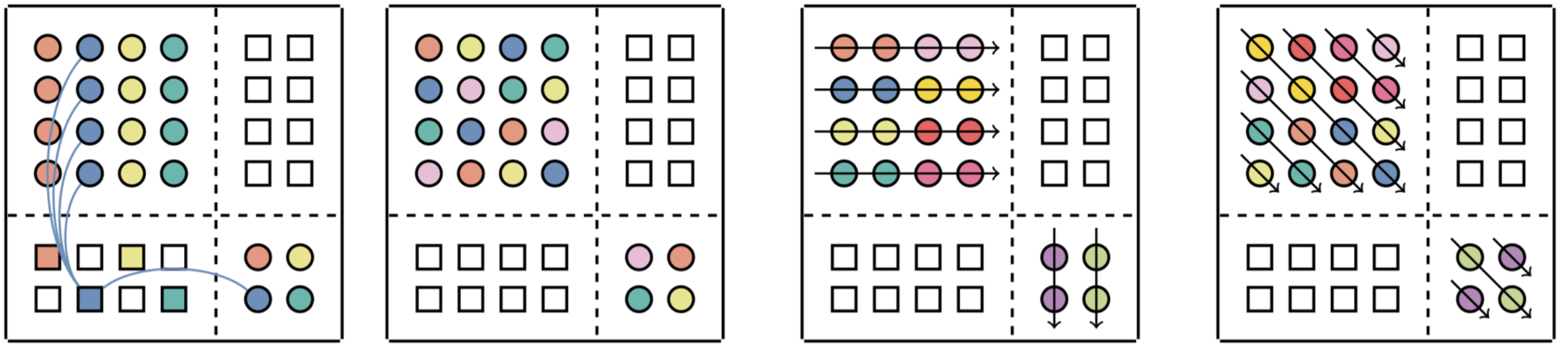
Error Recovery Failure (ERF)

- Decoding Failure (DF)
- Logical Error (LE)



[26] Nicholas Connolly, et al. "Fast erasure decoder for a class of quantum LDPC codes." *arXiv preprint arXiv:2208.01002* (2022).

Hypergraph Product (HGP) Codes



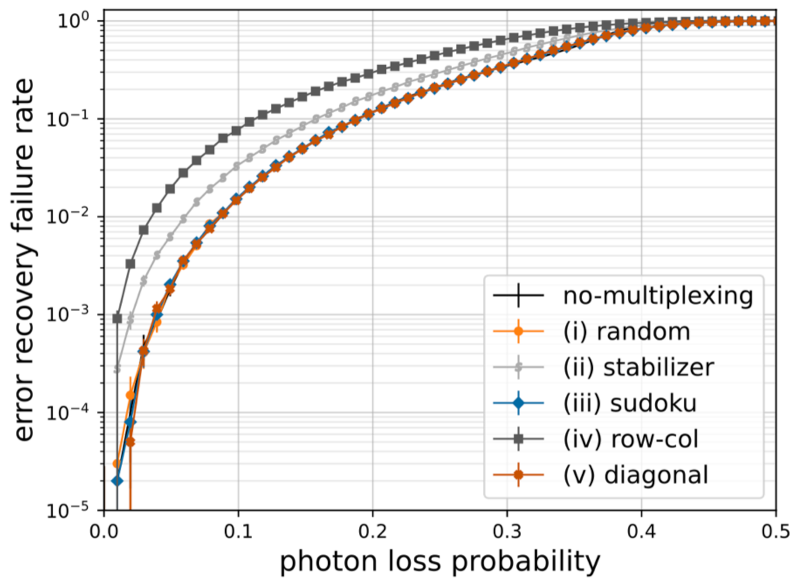
Strategy ii. Stabilizer | Strategy iii. Sudoku | Strategy iv. Row-Column | Strategy v. Diagonal

- Calculate distance is not as easy as surface codes
- Decoder-aware assignment strategies

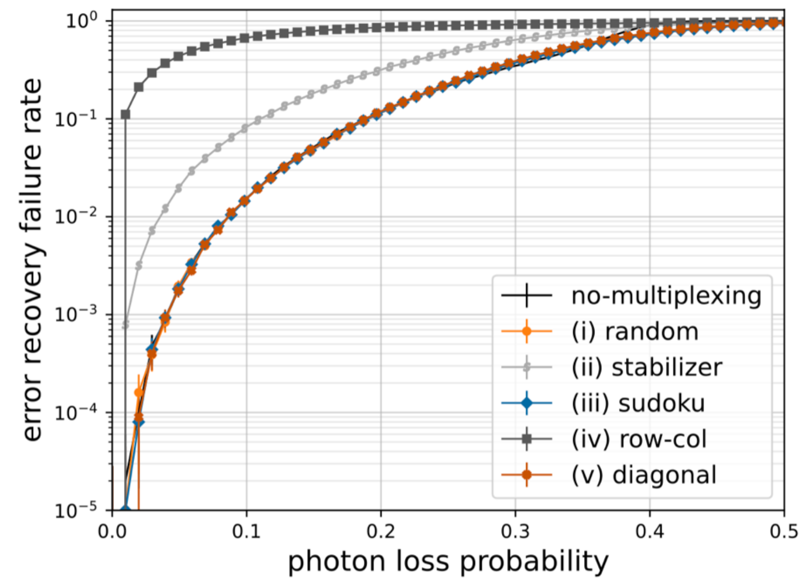
- Without proper placement, the effect of degradation increases a lot with m

Hypergraph Product (HGP) Codes

[[512, 8]] HGP codes $(16 \times 16, 16 \times 16)$



$m = 4$

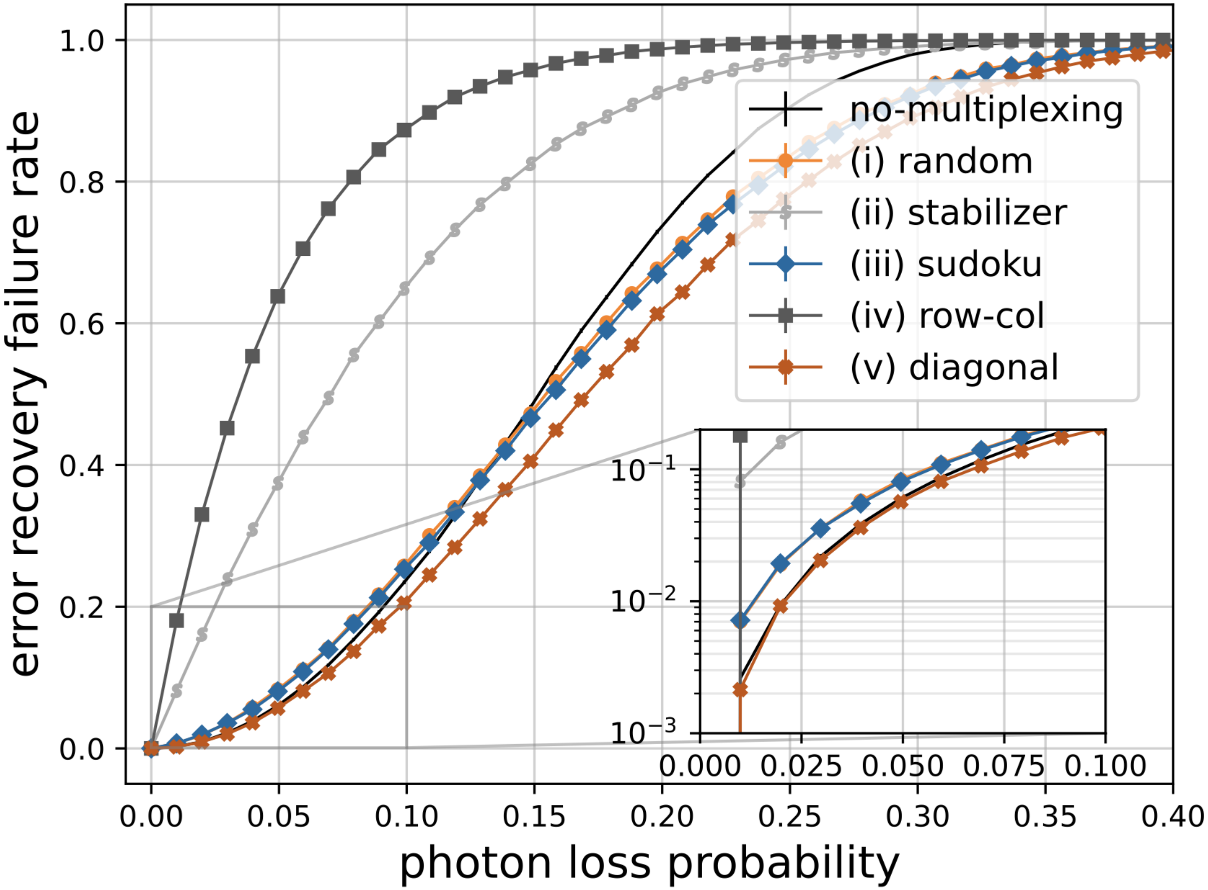


$m = 16$

(v) can achieve the same performance as without multiplexing while significantly reducing the resources required

Hypergraph Product (HGP) Codes

[[320, 82]] HGP codes $(16 \times 16, 8 \times 8)$ $m = 8$



(v) is even better than no-multiplexing!

Summary

arXiv:**2406.08832**

👍 QM reduce required number of resouces

- Number of gates → fast encoding
- Number of photon → Increase the throughput

QM may introduce a correlation on errors in qubits

- It may increase the logical error rate

↑could be dealt with assignment strategies

- Code structure aware (Surface codes)
- Decoder aware (HGP codes)

🧐

- Other assignment strategy
- Threshold behavior
- Efficiency of conversion of qubits to photons

arXiv:2302.00267

3-2 Programming Language and IR

Program analysis
and
optimization

High-level Programming Language

Lowering (Compiling)

Intermediate Representation

Abstraction

Quantum Architecture

Merits of formally defined Language

1. Allow us to discuss the behavior of programs
→ Program verification (guarantee that a program has some property during execution)

2. Modeling of computation in an appropriate way
→ More sophisticated program analysis, optimization, and intuitive structure of codes

3-2 Formal language for modular arch

arXiv:2302.00267

Setting

- Multiple quantum processors linked via quantum channels to realize scalable quantum computers
- Remote operations are achieved by using quantum channels

Compilation for circuit distribution

arXiv:2302.00267

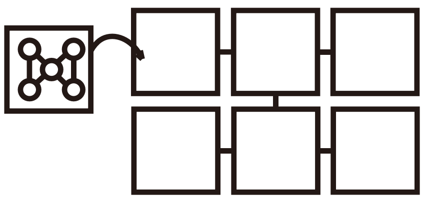
- Many quantum algorithms are designed for monolithic architectures
  - It is difficult to implement quantum circuits with existing frameworks (e.g. QMPI [27])

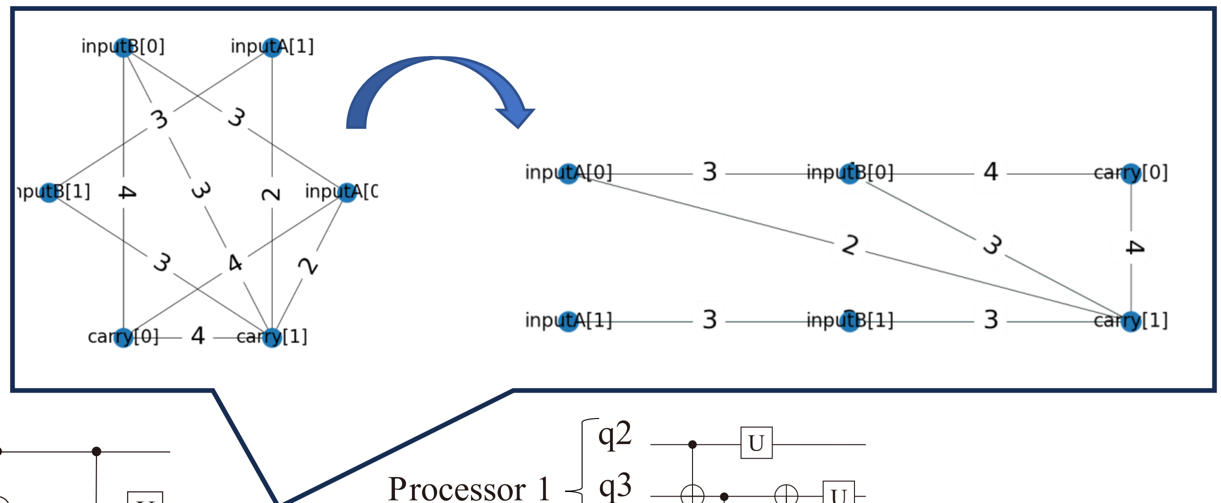
[27] T. Häner, D. S. Steiger, T. Hoefler, and M. Troyer, “Distributed quantum computing with QMPI”

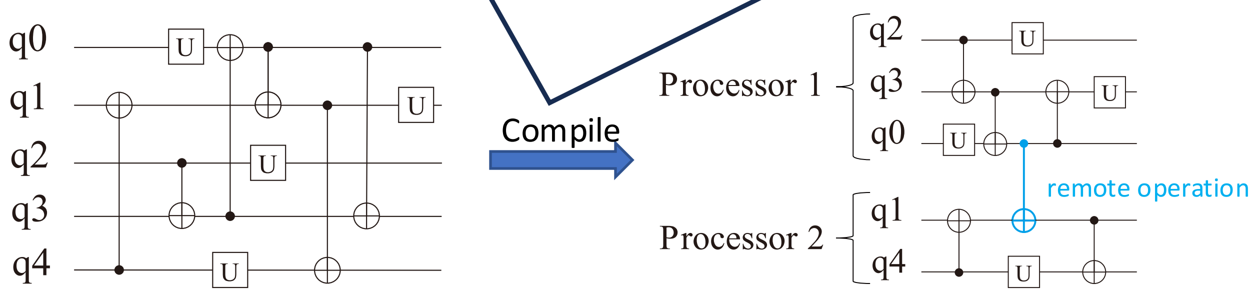
arXiv:2302.00267

Compilation for circuit distribution

- Distributed quantum compilers [28, 29] map quantum circuits to interconnected architectures







[28] D. Ferrari, A. S. Cacciapuoti, M. Amoretti, and M. Caleffi, "Compiler design for distributed quantum computing"
[29] O. Daei, K. Navi, and M. Zomorodi-Moghadam, "Optimized Quantum Circuit Partitioning"

How to define a typed IR

1. Instruction set and syntax
2. Semantics
3. Type system

Universality
Physical constraints
High-level or low-level?

How does the program behave?
Operational semantics,

What kind of property would be required for “safe execution”?

InQuIR: programming language

Flow

1. Define the syntax

$$
\begin{aligned}
e ::= {} & x = \texttt{init}() \\
& \mid U(x_1, \ldots, x_n) \\
& \mid x = \texttt{meas}(x_1, \ldots, x_n) \\
& \mid x = \texttt{genEnt}\ p \\
& \mid \texttt{entSwap}\ x_1\ x_2 \\
& \mid \texttt{QSEND}\ x_1\ \texttt{via}\ x_2 \\
& \mid x_1 = \texttt{QRECV}\ \texttt{via}\ x_2 \\
& \mid \texttt{RCXC}\ x_1\ \texttt{via}\ x_2 \\
& \mid \texttt{RCXT}\ x_1\ \texttt{via}\ x_2 \\
U ::= {} & X \mid Y \mid Z \mid CX \mid \ldots
\end{aligned}
$$

2. Define (operational) semantics

How the runtime state transitions

e.g.

$$
\frac{\forall v_i \in \mathit{DataQubit} \cup \mathit{CommQubit}}{[\rho, Q, E, [\![U(v_1, .., v_n); P]\!]_p, H] \to [U_{v_1, .., v_n} \rho U^{\dagger}_{v_1, .., v_n}, Q, E, [\![P]\!]_p, H]}
$$

3. Fast static analysis with type system!

$$
\frac{N+1 \mid \Gamma \vdash e}{N \mid \Gamma, x : \texttt{qbit} \vdash \texttt{QSEND}\ x\ \texttt{via}\ n; e}
\qquad
\frac{N > 0 \qquad N-1 \mid \Gamma, x : \texttt{qbit} \vdash e}{N \mid \Gamma \vdash x = \texttt{QRECV}\ \texttt{via}\ n; e}
$$

Typing rules for qubit usage analysis

Programming language for modular architecture

Introducing a quantum programming language to distributed quantum compilers has several advantages:

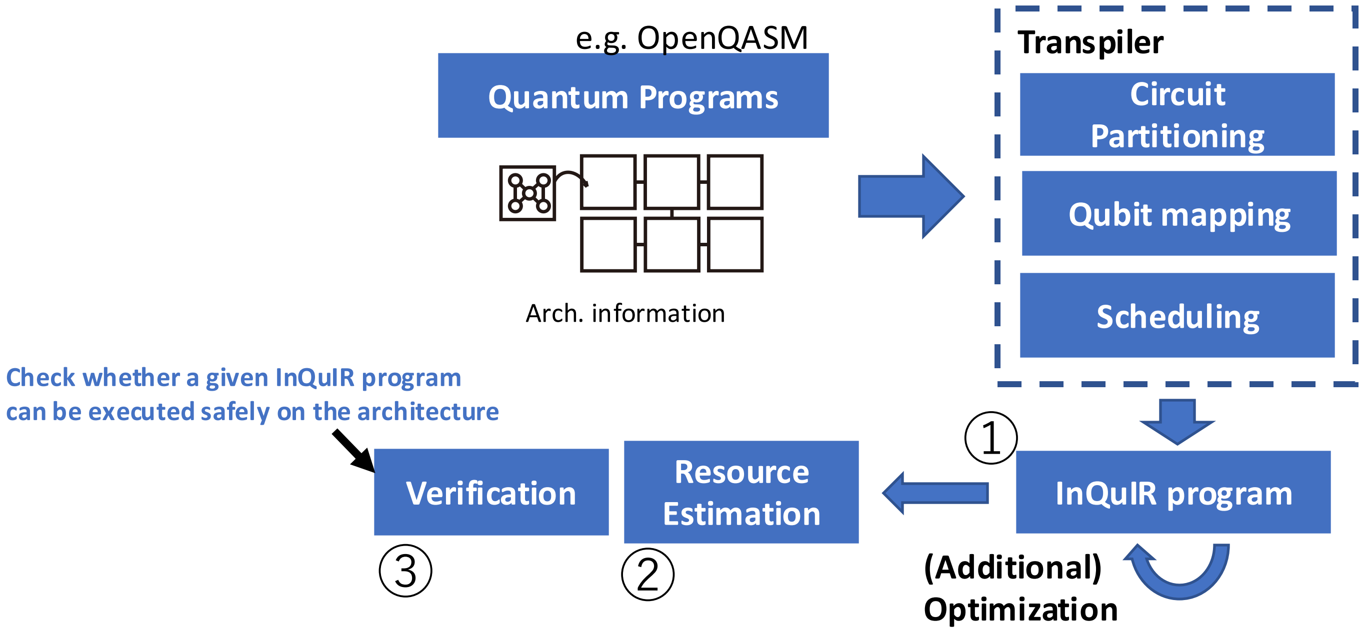
- The semantics of quantum programs are formally defined,
  so resource consumption can be estimated precisely

- Undesirable behaviors can be detected by using methods of
  static program analysis (e.g. type systems, abstract interpretation)

- Different quantum compilers can use the same program format

Contribution: InQuIR

- We propose InQuIR, an Intermediate Representation for Interconnected Quantum Computers (First formal language for distributed QC)



The (Brief) Syntax of InQuIR

$$
\begin{aligned}
e ::= {} & x = \mathtt{init}() \\
| \ & U(x_1, \ldots, x_n) \\
| \ & x = \mathtt{meas}(x_1, \ldots, x_n) \\
| \ & x = \mathtt{genEnt}\ p \\
| \ & \mathtt{entSwap}\ x_1\ x_2 \\
| \ & \mathtt{QSEND}\ x_1\ \mathtt{via}\ x_2 \\
| \ & x_1 = \mathtt{QRECV}\ \ \mathtt{via}\ x_2 \\
| \ & \mathtt{RCXC}\ x_1\ \mathtt{via}\ x_2 \\
| \ & \mathtt{RCXT}\ x_1\ \mathtt{via}\ x_2 \\
U ::= {} & X \mid Y \mid Z \mid CX \mid \ldots
\end{aligned}
$$

Each processor p has a sequence of instructions $\overrightarrow{e_p}$ to execute

Generate an entanglement with a processor p, and give it a label x

Sending/Receiving a qubit data by quantum teleportation consuming an entanglement x_2

Applying a Remote CX gate (in the Control/Target side) consuming an entanglement x_2

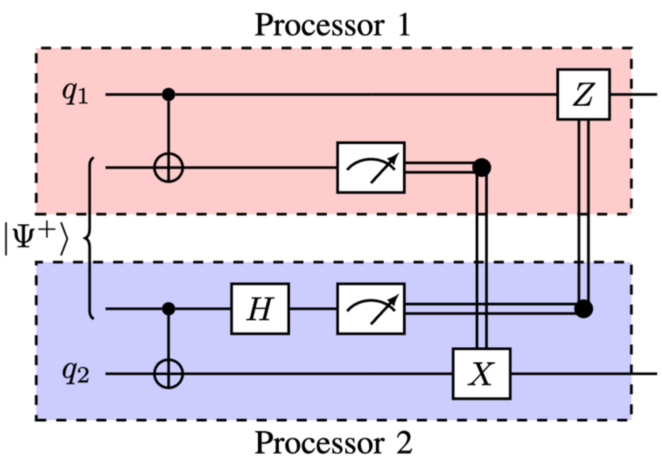
Applying CX gates remotely in InQuIR

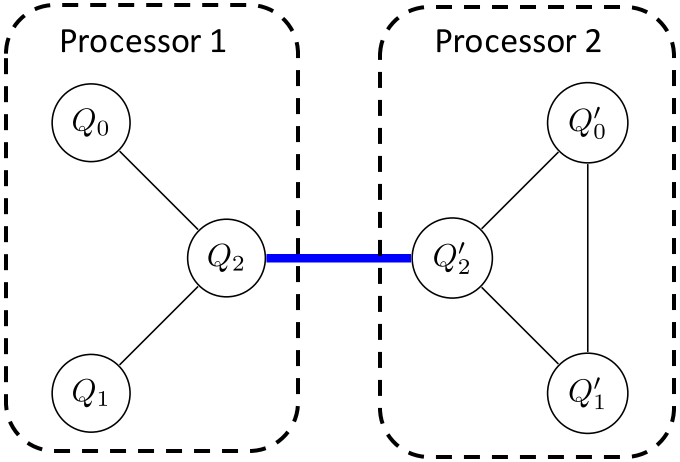
arXiv:2302.00267

Concurrent Processes

Processor 1 (p1):
```
q0, q1 = init();
q2 = genEnt p2;
RCXC q0 via q2;
q2 = genEnt p2;
QSEND q1 via q2;
```

Processor 2 (p2):
```
q0', q1' = init();
q2' = genEnt p1;
RCXT q0' via q2';
q2' = genEnt p1;
q1' = QRECV via
q2';
CX q0' q1';
```





Two (toy) Compilation Strategies

arXiv:2302.00267

Processor 1 (p1):

```
q0, q1 = init();
q2 = genEnt p2;
RCXC q0 via q2;
q2 = genEnt p2;
QSEND q1 via q2;
```

Processor 2:

```
q0’, q1’ = init();
q2’ = genEnt p1;
RCXT q0’ via q2’;
q2’ = genEnt p1;
q1’ = QRECV via
q2’;
CX q0’ q1’;
```

1. RCX strategy
   - Use RCXC/RCXT
2. MOVE strategy
   - Use QMOVE/QRECV

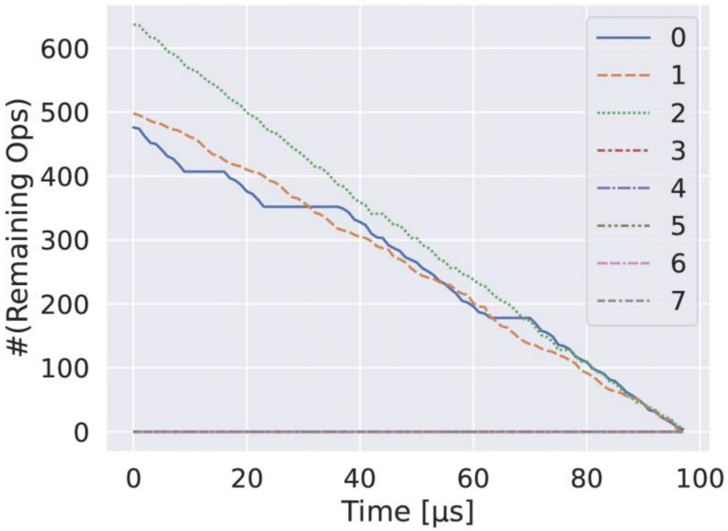
- #(communication depth) [30]

circuit name	RCX strategy	MOVE strategy
adder_63	304	334
life_238	5026	7220

[30] D. Ferrari, A. S. Cacciapuoti, M. Amoretti, and M. Caleffi, “Compiler design for distributed quantum computing”

arXiv:2302.00267

Network utilization analysis and visualization



Evaluation of compilation strategies
by the same indicators (e.g. # of instructions, depth)

Estimating the hardware topology required for apps
→provide guidelines for hardware design

Future works?

- Introduction of new metrics for evaluation
- QEC and FT logics
- Find the "bugs" you want to verify with type system
- (Implementations...)

Circuit name	N	E-count	C-count	Linear: $(2,2)\times 8$			Linear: $(2,4)\times 8$			Linear: $(2,6)\times 8$		
				D_E	D_C	cost[ns]	D_E	D_C	cost	D_E	D_C	cost [ns]
adr4_197	16	5308	10616	1020	3150	1562850	510	2248	751630	510	2248	751630
ising_model_16	16	140	280	10	20	13510	5	20	7280	5	20	7280
rd53_138	16	122	244	33	74	47730	17	66	23610	17	66	23610
sqn_258	16	15054	30108	2843	9606	4393910	1365	6024	2136340	1365	6024	2136340
root_255	16	31286	62572	5112	19268	8086560	2596	12878	3942970	2596	12878	3942970
4gt12-v1_89	16	224	448	68	136	97370	34	118	48780	34	118	48780
9symml_195	16	66732	133464	10512	40366	16504980	5342	25616	8025860	5342	25616	8025860
life_238	16	42796	85592	6755	26076	10628990	3432	16564	5153840	3432	16564	5153840

Verification of InQuIR

- We can statically check the “safety” of InQuIR programs by static program analysis
  - e.g. the qubit exhaustion and deadlocks do not occur

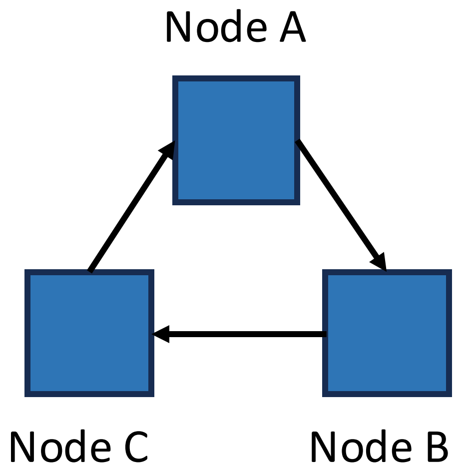
- A type system can be formalized to analyze qubit utilization

$$\frac{N+1 \mid \Gamma \vdash e}{N \mid \Gamma, x : \mathtt{qbit} \vdash \mathtt{QSEND}\ x\ \mathtt{via}\ n; e} \qquad \frac{N > 0 \qquad N-1 \mid \Gamma, x : \mathtt{qbit} \vdash e}{N \mid \Gamma \vdash x = \mathtt{QRECV}\ \mathtt{via}\ n; e}$$

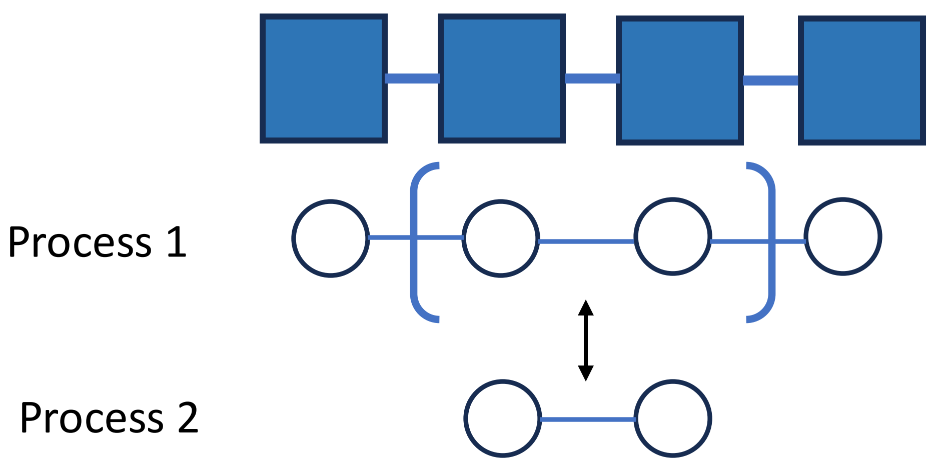
Typing rules for qubit usage analysis

Verification of InQuIR

- Case study: deadlock free



Waiting relation is cyclic
→ stack!



Path blocking for entanglement swapping

Ongoing & Future Work

- Extending primitive operations of InQuIR to enable sophisticated optimizations
  - RCXC/RCXT should be decomposed to enable quasi-parallelism [31]
- Dealing with nondeterministic protocol
  - Entanglement purification / feed-forward
- Seeking new metrics to monitor resource usage

[31] D. Cuomo, M. Caleffi, K. Krsulich, F. Tramonto, G. Agliardi, E. Prati, and A. S. Cacciapuoti, "Optimized compiler for distributed quantum computing"

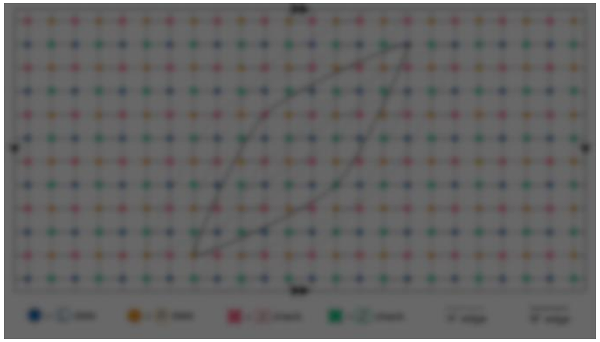
4. Towards large-scale quantum computer

For the implementation of useful & scalable FTQC

- Topological codes → High-rate LDPC codes

符号	k	d	degree
2D surface codes	$O(1)$	$n^{\frac{1}{2}}$	4
2D hyperbolic surface codes	$\Omega(n)$	$\log(n)$	$O(1)$
3D surface codes	$O(1)$	$(n\ \log n)^{\frac{1}{2}}$	$O(1)$
Hypergraph product codes	$\Omega(n)$	$n^{\frac{1}{2}}$	$O(1)$

Bivariate bicycle



Bravyi, Sergey, et al. "High-threshold and low-overhead fault-tolerant quantum memory." *Nature* 627.8005 (2024): 778-782.

4. Towards large-scale quantum computer

overhead for magic state distillation scales as $O(\log^{\gamma}(1/\epsilon))$

~~Open Question~~

Is there any quantum error correction codes with $\gamma \to 0$

where $\gamma = \log_d \frac{n}{k}$?

Yes! There exist![32-35]

[32] Adam Wills, Min-Hsiu Hsieh, and Hayata Yamasaki. Constant-Overhead Magic State Distillation, August 2024. arXiv:2408.07764 [quant-ph]. ← 8/14

[33] Louis Golowich and Venkatesan Guruswami. Asymptotically Good Quantum Codes with Transversal Non-Clifford Gates, August 2024. arXiv:2408.09254 [quant-ph]. ← 8/17

[34] Quynh T. Nguyen. Good binary quantum codes with transversal CCZ gate, August 2024. arXiv:2408.10140 [quant-ph]. ← 8/19

[35] Thomas R. Scruby, Arthur Pesah, and Mark Webster. Quantum Rainbow Codes, August 2024. arXiv: 2408.13130 [quant-ph] ← 8/23

4. Towards large-scale quantum computer

Quantum Error Correction Codes & Hardware

↓

Logical Gate Implementation & Microarchitecture

↓

Domain-specific Language

↓

Compilation, Resource Analysis, and Verification

↓

Architecture

Summary

Hardness of braided quantum circuit optimization in the surface code

- Kunihiro Wasa, **Shin Nishio**, Koki Suetsugu, Michael Hanks, Ashley Stephens, Yu Yokoi, Kae Nemoto
- IEEE Transactions on Quantum Engineering vol. 4, pp. 1-7, 2023
- preprint: arXiv[quant-ph] 2302.00273

InQuIR: Intermediate Representation for Interconnected Quantum Computers

- **Shin Nishio**, Ryo Wakizaka
- preprint: arXiv[quant-ph] 2302.00267

Resource Reduction in Multiplexed High-Dimensional Quantum Reed-Solomon Codes

- **Shin Nishio**, Nicolò Lo Piparo, Michael Hanks, William John Munro, Kae Nemoto
- Physical Review A 107, 032620
- preprint: arXiv[quant-ph] 2206.03712

Multiplexed Quantum Communication with Surface and Hypergraph Product Codes

- **Shin Nishio**, Nicholas Connolly, Nicolò Lo Piparo, William John Munro, Thomas Rowan Scruby, Kae Nemoto
- preprint: arXiv[quant-ph] 2406.08832

Summary

- Quantum computing systems include the classical computer systems
  - Defect braiding circuit optimization Phys. Rev. A **107**, 032620

- Modular architecture may be an efficient implementation of large-scale FTQC systems
  - Multiplexing can reduce resources for communication Phys. Rev. A **107**, 032620, arXiv:**2406.08832**
  - Formal language and intermediate representation play an important role in program optimization and analysis arXiv:**2302.00267**

Thank you!

CSS Code [16,17]

$\mathcal{C}_1 = [n, k_1, d_1]$ [physical, logical, min distance] dimension
$\mathcal{C}_2 = [n, k_2, d_2]$ $(\{0\} \subset \mathcal{C}_2 \subset \mathcal{C}_1 \subset \mathbb{F}_2^n, k_2 < k_1)$

It is possible to define coset of $\mathcal{C}_2$ about $\mathcal{C}_1$ since $\mathcal{C}_2 \subset \mathcal{C}_1$.
Especially $\mathcal{C}_w = \{v + w | v \in \mathcal{C}_2\}$ is a set which separates $\mathcal{C}_2$ into different cosets.

→Each $\mathcal{C}_w$ can choose unique set of $k_1 - k_2$ vectors in $\mathcal{C}_1$
(Coset representative)

Suppose v is a coset representative. The $[[n, k_1 - k_2]]$codeword is given as

$$|v\rangle = \frac{1}{\sqrt{2_2^k}} \sum_{w \in C_2} |w + v\rangle$$

[16] A. Robert Calderbank, and Peter W. Shor. "Good quantum error-correcting codes exist." *Physical Review A* 54.2 (1996): 1098.
[17] Andrew Steane, "Multiple-particle interference and quantum error correction." *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 452.1954 (1996): 2551-2577.

Introduction to QRS : Galois Field

Galois Field is a field with finite number of elements. Addition /Multiplication for the element of $GF(p)$ is defined as modulo p addition/multiplication where p is a prime number

e.g. $GF(5) = \{0, \alpha, \alpha^2, \alpha^3, 1\}$
Suppose $\alpha = 2$, then $GF(5) = \{0,2,4,3,1\}$

+	0	2	4	3	1
0	0	2	4	3	1
2	2	4	1	0	3
4	4	1	3	2	0
3	3	0	2	1	4
1	1	3	0	4	2

×	0	2	4	3	1
0	0	0	0	0	0
2	0	4	3	1	2
4	0	3	1	2	4
3	0	1	2	4	3
1	0	2	4	3	1

Expansion of Galois Field

It is possible to define Galois extension $GF(p^k)$ by using minimal polynomial (monic k-dim polynomial of $GF(p)$).

The root of the minimal polynomial is called primitive element and denote it by α.

e.g. $GF(2) = \{0,1\}$

$GF(2^2) = \{0,1,\alpha,\alpha^2\}$

$f(x) = x^2 + x + 1$ minimal polynomial

$+$	0	1	α	α^2
0	0	1	α	α^2
1	1	0	α^2	α
α	α	α^2	0	1
α^2	α^2	α	1	0

$\times$	0	1	α	α^2
0	0	0	0	0
1	0	1	α	α^2
α	0	α	α^2	1
α^2	0	α^2	1	α

Appendix: SUM Gate for $d = 5$

RCA part

$$\overset{4\,2\,1}{|abc\rangle}\,\overset{4\,2\,1}{|def\rangle} \qquad \overset{\text{carry}}{\overset{8\,4\,2}{|000\rangle}} \qquad \overset{\text{check if}}{\overset{7\,6\,5}{|000\rangle}}$$

$$|abc\rangle\,|(a+\overset{4}{d}+c4)(b+\overset{2}{e}+c2)(c+\overset{1}{f})\rangle\,|000\rangle$$

if a=d==1:
carry 8
(overflow)

if b=e==1:
carry 4

if c=f==1:
carry 2

Full adder part
(written in mod 8)

Modulo part

do nothing for

$0 \equiv 0\ mod\ 5$
$1 \equiv 1\ mod\ 5$
$2 \equiv 2\ mod\ 5$
$3 \equiv 3\ mod\ 5$
$4 \equiv 4\ mod\ 5$

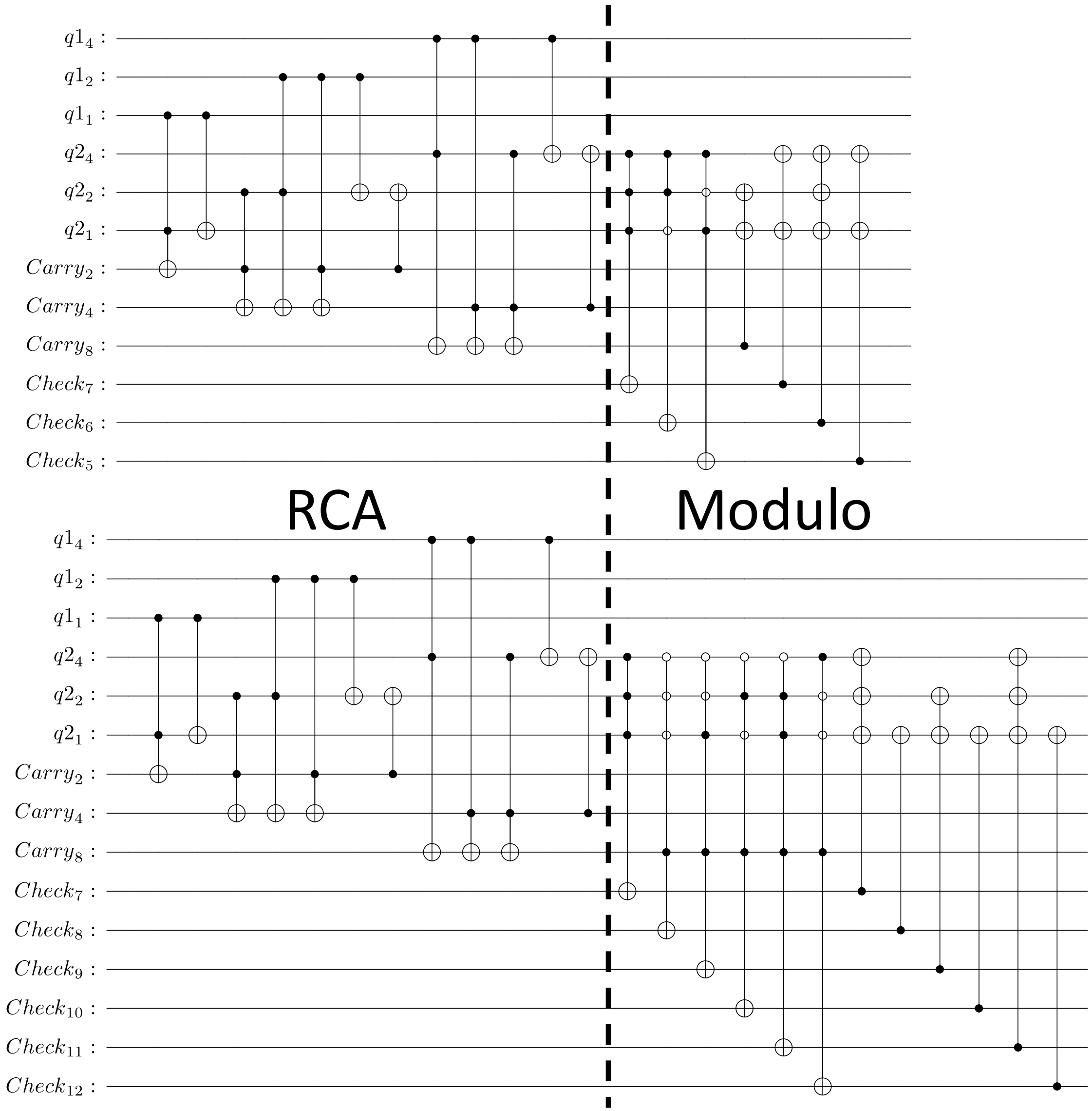
check if (activate anscillae)

$5 \equiv 0\ mod\ 5$
$6 \equiv 1\ mod\ 5$
$7 \equiv 2\ mod\ 5$
$8 \equiv 3\ mod\ 5$

Appendix: Implementation of SUM gate

mod 5

mod 7



- $2(d-1) \leq 2^k$

- $2(d-1) > 2^k$

need to check the overflow
→ require $C_{k+1}X$ gate

Appendix: $GF(2^m)$ QRS Code

- Required gates for $GF(2^m)$ QRS code can be implemented using only CX gates
- A more efficient way to implement this using Toffoli has not been found.

e.g. $GF(4)$

$$GF(4) = \{0, 1, \alpha, \alpha^2\} = \{0, 1, \alpha\ , \alpha + 1\} = \{00, 01, 10, 11\}$$

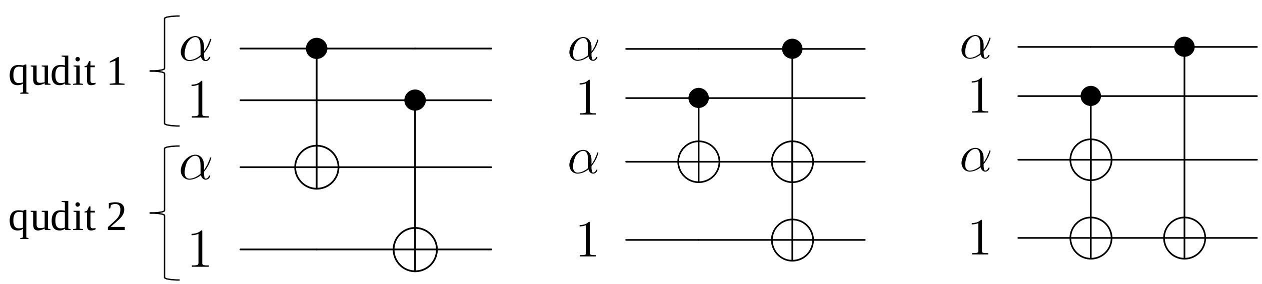
exponential polynomial vector

$x^2 + x + 1 = 0$

primitive polynomial

Required gates

- $C1$ gate
  $|a\rangle|b\rangle \rightarrow |a\rangle|a + b\rangle$
- $C\alpha$ gate
  $|a\rangle|b\rangle \rightarrow |a\rangle|a\alpha + b\rangle$
- $C\alpha^2$ gate
  $|a\rangle|b\rangle \rightarrow |a\rangle|a\alpha^2 + b\rangle$



$C1$ gate $C\alpha$ gate $C\alpha^2$ gate

general decomposition m CX $m - 1 + H_w(\alpha^m)$ CX

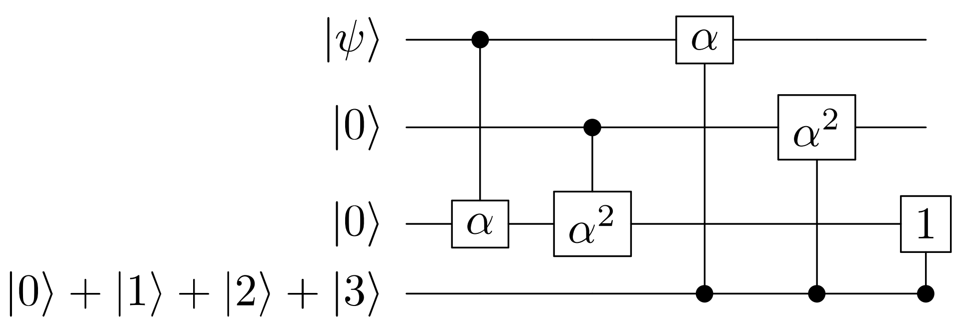
e.g. $GF(2^m)$ QRS Code: $[[3,1,3]]_5$ code

$\mathcal{C}_1 = [3,2,2]_4 \ \mathcal{C}_2 = [3,1,3]_4$

generator matrix and parity check matrix

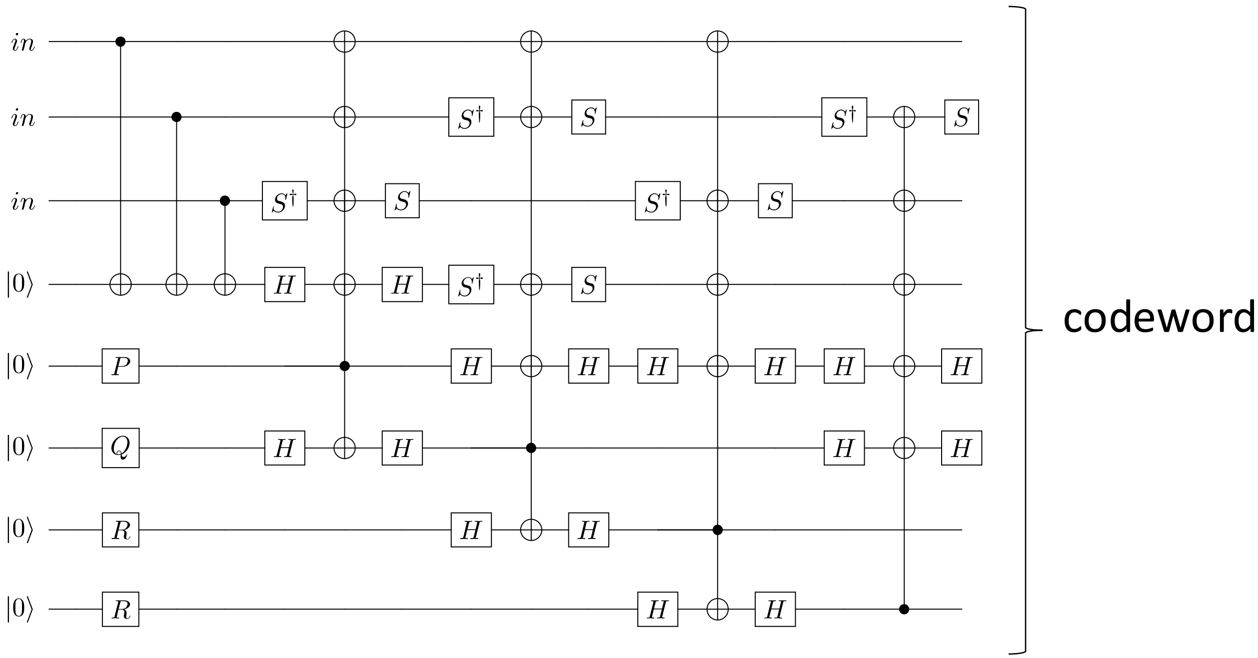
$$g(x) = (x-1)(x-\alpha) = x^2 + (1+\alpha)x + (\alpha)$$

$$G = \begin{pmatrix} 1 & 0 & \alpha \\ 0 & 1 & \alpha^2 \end{pmatrix} \qquad H = \begin{pmatrix} \alpha & \alpha^2 & 1 \end{pmatrix}$$



Appendix: Stabilizer Codes

- Efficient encoding circuit for general stabilizer codes require many controlled gates with multi-targets[19].
- Cannot apply our method directly

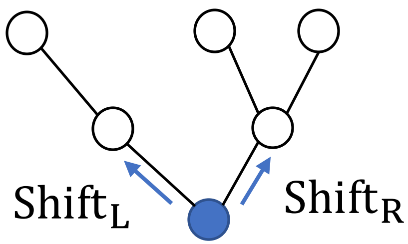


[19] Cleve, Richard, and Daniel Gottesman. "Efficient computations of encodings for quantum error correction." *Physical Review A* 56.1 (1997): 76.

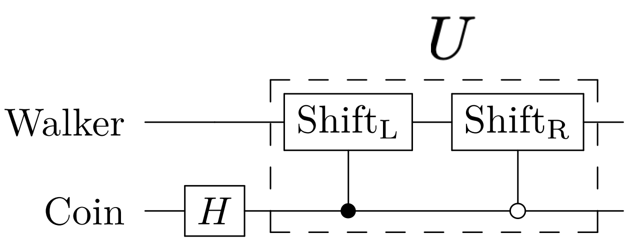
Other Applications

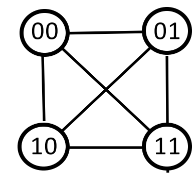
Discrete time quantum walk algorithm[15]

DTQW on a graph[16]

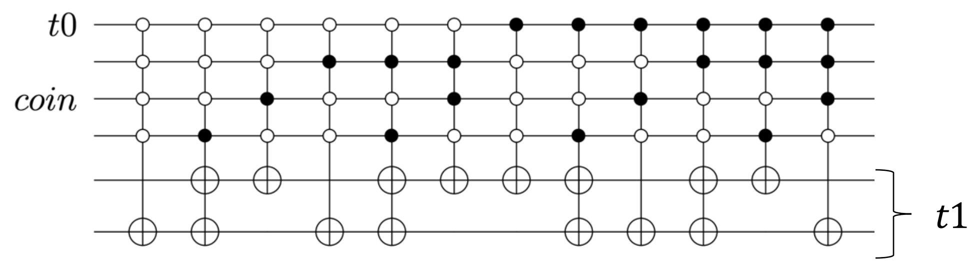


time evolution $|\psi_t\rangle = U^t\,|\psi_0\rangle$





e.g. quantum walk on fully connected graph



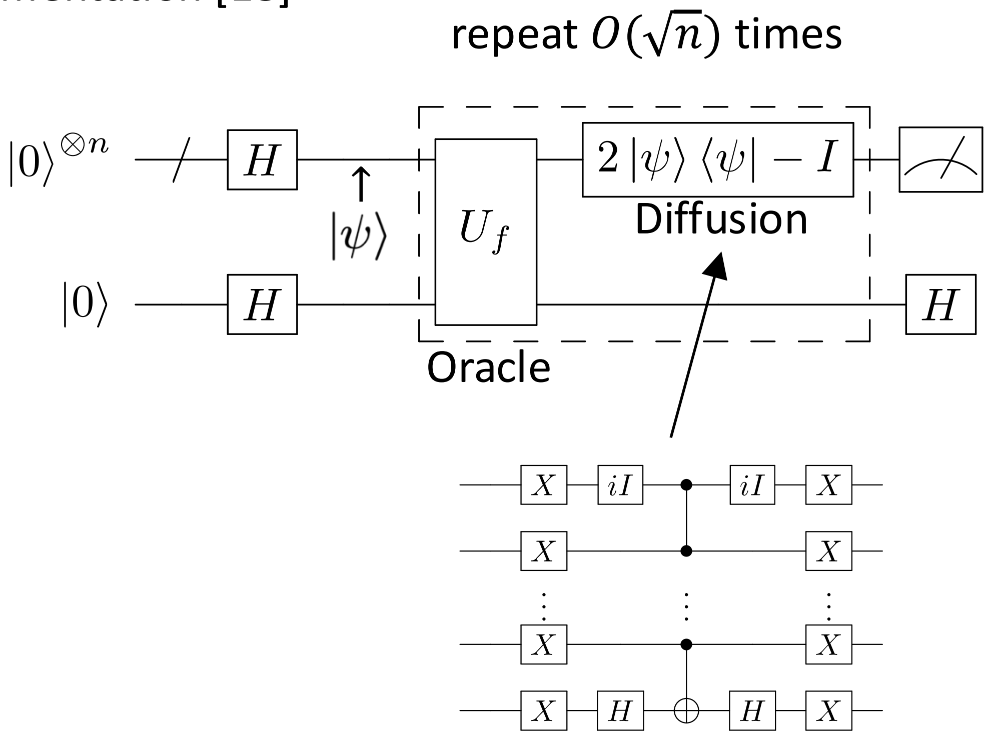
[15] Y. Aharonov, L. Davidovich, and N. Zagury. Quantum random walks. Physical Review A, 48(2):1687, 1993.

[16] B. Douglas and J. Wang. Efficient quantum circuit implementation of quantum walks. Physical Review A, 79(5):052335, 2009.

Other Applications

Grover’s search[17]

Circuit Implementation [18]



[17] L. K. Grover. A fast quantum mechanical algorithm for database search. In Proceedings of the twenty- eighth annual ACM symposium on Theory of com- puting, pages 212–219, 1996.
[18] C. Lavor, L. Manssur, and R. Portugal. Grover’s al- gorithm: Quantum database search. arXiv preprint quant-ph/0301079, 2003.

arXiv:2302.00267

Applying CX gates remotely in InQuIR

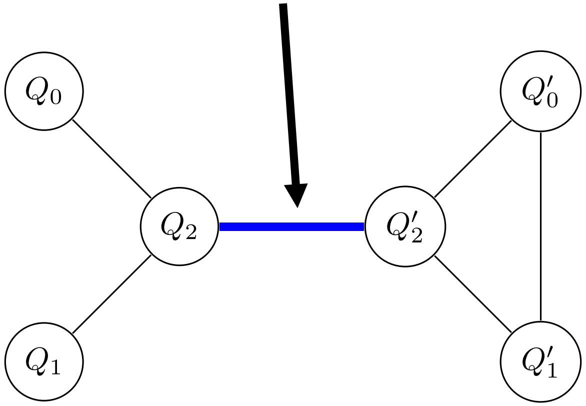
Processor 1 (p1):

```
q0, q1 = init();
q2 = genEnt p2;
RCXC q0 via q2;
q2 = genEnt p2;
QSEND q1 via q2;
```

Processor 2 (p2):

```
q0', q1' = init();
q2' = genEnt p1;
RCXT q0' via q2';
q2' = genEnt p1;
q1' = QRECV via
q2';
CX q0' q1';
```

Creating entanglement between Q_2 and Q_2'



arXiv:2302.00267

Applying CX gates remotely in InQuIR

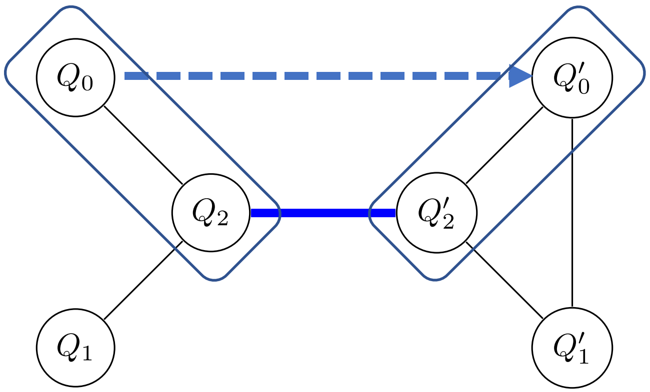
Processor 1 (p1):

```
q0, q1 = init();
q2 = genEnt p2;
RCXC q0 via q2;
q2 = genEnt p2;
QSEND q1 via q2;
```

Processor 2 (p2):

```
q0', q1' = init();
q2' = genEnt p1;
RCXT q0' via q2';
q2' = genEnt p1;
q1' = QRECV via
q2';
CX q0' q1';
```

Apply a remote CX gate to (Q_0, Q'_0) by consuming the entangle pair (Q_2, Q'_2), where Q_0 is a control qubit



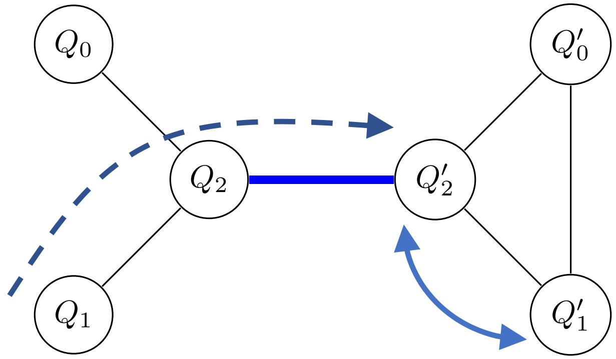
This procedure does not change the position of data qubits

Applying CX gates remotely in InQuIR

Processor 1 (p1):
```
q0, q1 = init();
q2 = genEnt p2;
RCXC q0 via q2;
q2 = genEnt p2;
QSEND q1 via q2;
```

Processor 2:
```
q0', q1' = init();
q2' = genEnt p1;
RCXT q0' via q2';
q2' = genEnt p1;
q1' = QRECV via
q2';
CX q0' q1';
```

Teleport Q_1 to Q_2', and then move the data to Q_1' from Q_2'



Appendix

Algorithm 1: Strategy iv. random + threshold

Input: $P = \{p_i\}$ (the set of photons) where initially $p_i = \{\emptyset\}$ (the set of qubits to be encoded in the i^{th} photon), $Q = \{q_j\}$ (the list of all physical qubits in the code), and the number m of qubits in a single photon.

Output: $P = \{p_i\}$ (set of set of qubits in i^{th} photon).

Initialize the threshold with $T := \frac{d}{2} - 1$;
for *photon* $p_i \in P$ **do**
    Pick a qubit $q_j \in Q$ randomly.;
    Move q_j from Q to p_i;
    while $|p_i| < m$ **do**
        while $|p_i| < m$ *and* $Q \neq \emptyset$ **do**
            Pick a candidate qubit $q_k \in Q$ randomly;
            if q_k *has minimum distance greater than* T *from all the qubits in* p_i **then**
                Move q_k from Q to p_i;
            else
                Move q_k from Q to a waiting list Q';
        Move all qubits in Q' to Q;
        Update $T := T - 1$;
Return P;

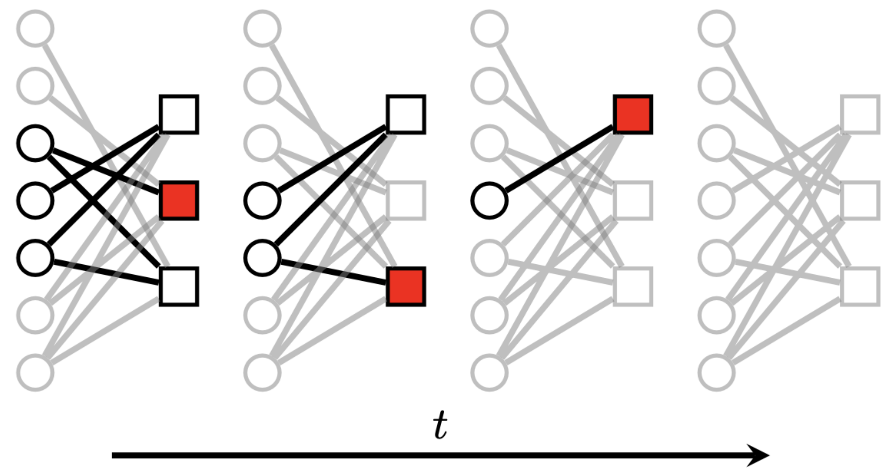
Appendix

Algorithm 2: Peeling Algorithm

Input: A code $\mathrm{Ker}(H)$ with Tanner graph G, a set of erased bits E, and a syntax syndrome vector s.

Output: A predicted error $\hat{e} \subseteq E$ such that $H\hat{e} = s$, or **Failure**.

Initialize $\hat{e} = \emptyset$;
while $E \neq \emptyset$ **do**
    Compute erasure subgraph $G_E \subseteq G$;
    if $\exists$ *dangling check* $s_i \in G_E$ **then**
        if s_i *is unsatisfied* **then**
            Error on adjacent bit $b_j \in E$;
            Flip bit b_j, update syndrome s;
            Update $\hat{e} := \hat{e} \cup \{b_j\}$;
        else
            No error on adjacent bit b_j;
        Update $E := E \setminus \{b_j\}$;
    else
        Return **Failure**;
Return $\hat{e}$;



$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Appendix

Algorithm 3: Strategy iii. sudoku

Input: $P = \{p_i\}$ (the set of photons, where p_i is the set of qubits in photon i), $Q = \{q_j = (r_j, c_j, b_j)\}$ (a list of 3-tuples with the row, column, and block of each physical qubit in the HGP code), and the number m of qubits per photon.

Output: $P = \{p_i\}$ (photon assignments).

for *photon* $p_i \in P$ **do**
  Pick a qubit $q_j \in Q$ randomly;
  Move q_j from Q to p_i;
  while $|p_i| < m$ *and* $Q \neq \emptyset$ **do**
    Pick a candidate qubit $q_k \in Q$;
    if q_k *is in a different row and column (or block) from each previously selected* $q_j \in p_i$ $((r_k \neq r_j$ *and* $c_k \neq c_j)$ *or* $b_k \neq b_j)$ **then**
      Move q_k from Q to p_i;
    else
      Move q_k from Q to a temporary waiting list Q';
  Move all qubits in Q' back to Q;
  while $|p_i| < m$ **do**
    Pick a qubit $q_k \in Q$ randomly;
    Move q_k from Q to p_i;
Return P;

Algorithm 4: Strategy v. diagonal

Input: $P = \{p_i\}$ (the set of photons where p_i is the set of qubits in photon i), $Q = \{q_j\}$ (a list of physical qubits ordered along the diagonal), and the number m of qubits per photon.

Output: $P = \{p_i\}$ (photon assignments).

for *photon* $p_i \in P$ **do**
  for *qubits with indices* $j \in \{im, \cdots, (i+1)m\}$ **do**
    Move q_j from Q to p_i
return P;