Quantum Error Correction Near the Hashing Bound with Linear-Time Decoding in Code Length

Kenta Kasai

Institute of Science Tokyo

Quantum Error Correction Theory Workshop for Young Researchers
December 17–19, 2025

Classical Coding Theory: A Completed Success

- Classical coding theory has reached a highly mature stage, achieving channel capacity with BP decoding and complexity linear in the block length.
- The natural question is whether these powerful classical principles can be carried over to quantum error correction.

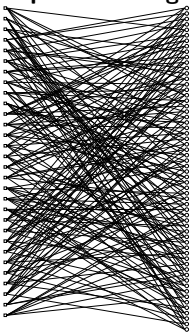
What We Have Learned from Classical LDPC Codes

- **BP decoding** exploits the sparsity of the parity-check matrix H and iteratively updates the reliability of each bit using local parity constraints.
- **Density evolution** reveals that regular degree distributions of H already yield excellent BP performance in the waterfall region.
- Imposing a **column weight** ≥ 3 in *random sparse codes* guarantees linear growth of the minimum distance.
- The **girth** of the Tanner graph plays a central role in both decoding dynamics and distance properties.
- Non-binary LDPC codes over **finite field extensions** achieve strong BP performance even with column weight two.
- Ultimately, the minimum distance determines the error-floor behavior.

Example: $(3, 6)$-regular Code

$H =$

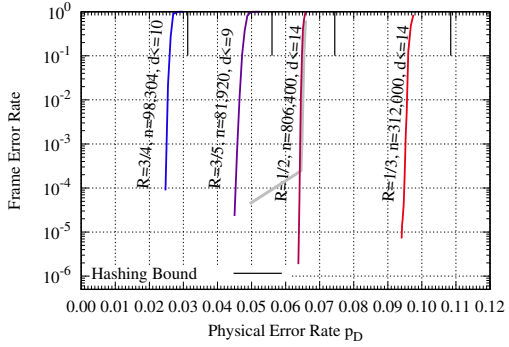
Example: Tanner graph



Numerical Results: Qubit Codes Based on Non-Binary LDPC Codes

Let us first present the decoding performance of the proposed scheme. The following performance has been achieved.

- A clear **threshold-like** FER curve is observed, approaching the hashing bound.
- Belief propagation decoding has **linear complexity** in the number of logical qubits. No OSD or heavy post-processing is used.
- **No error floor** has been observed down to a frame error rate of at least $10^{-4[<35\triangle 9\triangle 16M}$.
- A remarkably high **coding rate** is achieved: $R \geq 1/3$.
- The number of logical qubits is close to $k = 10^6$, which is exceptionally large for quantum LDPC codes.
- A GPU implementation enables a decoding throughput of approximately **1M [qbps]**.



[a] https://github.com/kasaikenta/gd-css-decoder
[b] https://github.com/NagatsukiSep/gd-css-decoder

[a] Komoto and Kasai, *npj Quantum Information*, 2025.
[b] Kasai, Hagiwara, Imai, and Sakaniwa, IEEE Trans. IT, 2011.
[c] K. Kasai, *arXiv:2506.15636*, 2025.
[d] K. Kasai, IEEE ISTC 2025.

General Procedure of Error Correction

Classical Error Correction

1. A classical LDPC code is defined by a *single* parity-check matrix H.
2. Physical noise is described by a binary error vector $e \in \mathbb{F}_2^n$.
3. We measure the syndrome
$$s = He.$$
4. The decoder finds $\hat{e}$ such that
$$H\hat{e} = s.$$
5. **Success criterion.**
$$\hat{e} + e \in \{0\}.$$

Quantum Error Correction

1. Quantum LDPC codes require *two* matrices H_X, H_Z satisfying
$$H_X H_Z^{\mathsf{T}} = 0.$$
2. Physical noise is described by (x, z).
3. We measure
$$s_X = H_X z, \qquad s_Z = H_Z x.$$
4. The decoder finds $\hat{x}, \hat{z}$ such that
$$H_Z \hat{x} = s_Z, \quad H_X \hat{z} = s_X.$$
5. **Success criterion.**
$$\hat{x} + x \in C_Z^{\perp}, \ \hat{z} + z \in C_X^{\perp}.$$

General Construction of Quantum LDPC Codes

Quantum LDPC codes require *two* parity-check matrices, H_X and H_Z, which must satisfy the orthogonality condition

$$H_X H_Z^{\mathsf{T}} = 0 \quad \text{(over } \mathbb{F}_2\text{)}.$$

- This commutativity constraint tightly couples the design of H_X and H_Z, severely restricting the row and column weight distributions as well as the achievable girth.
- As a consequence, many constructions start from a pair of highly structured and fully orthogonal sparse *full* matrices, which typically yield zero quantum rate $R = (n - \mathrm{rank}(H_X) - \mathrm{rank}(H_Z))/n = 0$.

$$H_X^{(\mathrm{ful})} \big(H_Z^{(\mathrm{ful})}\big)^{\mathsf{T}} = 0.$$

- The final parity-check matrices H_X and H_Z are obtained by removing a subset of check rows from the full matrices in order to achieve the desired code rate.

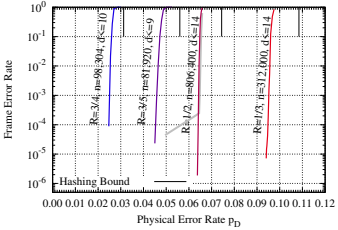
$$H_X^{(\mathrm{ful})} = \begin{bmatrix} H_X \\ H_X^{(\mathrm{res})} \end{bmatrix}, \qquad H_Z^{(\mathrm{ful})} = \begin{bmatrix} H_Z \\ H_Z^{(\mathrm{res})} \end{bmatrix}$$

Each row z of $H_X^{(\mathrm{res})}$, which typically has low weight, automatically satisfies $H_Z z^{\mathsf{T}} = 0$, and hence $z \in C_Z$. In general, $z \notin C_X^{\perp}$, so z acts as a Z-type *logical* operator in $C_Z \setminus C_X^{\perp}$.

Key Ingredients Behind the Proposed Quantum LDPC Codes

The proposed performance is achieved by combining the following key ingredients.

- We adopt a generalized **Hagiwara–Imai** construction.
- **Commutativity** between H_X and H_Z is precisely controlled using affine permutation matrices (APMs).
- For non-binary LDPC codes over finite-field, a column weight of $J = 2$ is optimal, enabling **ultra-sparse** parity-check matrices and guaranteeing a **large girth (≥ 12)**.
- When the column weight is $J = 2$, all codewords are generated by cycles or unions of cycles in the Tanner graph. By **carefully assigning non-binary symbols**, such cycle-based structures are prevented from forming valid codewords, thereby eliminating small logical errors.
- **Joint belief propagation** decoding is used to fully exploit correlations between X and Z errors.
- Dominant error events caused by trapping sets associated with length-12 cycles in the error-floor region are corrected by a **post-processing** algorithm with $O_n(1)$ complexity.



[a] Komoto and Kasai, *npj Quantum Information*, 2025.

[b] Kasai, Hagiwara, Imai, and Sakaniwa, IEEE Trans. IT, 2011.

[c] K. Kasai, *arXiv:2506.15636*, 2025.

[d] K. Kasai, ISTC 2025.

Generalized Hagiwara–Imai Construction: (J, L)-Regular LDPC CSS Codes

- The generalized Hagiwara–Imai construction provides a systematic method for constructing (J, L)-regular LDPC CSS codes by arranging permutation blocks in a cyclic manner.
- Let $\{F_i\}_{i=0}^{L/2-1}$ and $\{G_i\}_{i=0}^{L/2-1}$ be $P \times P$ permutation matrices.
- The full parity-check matrices $H_X^{(\mathrm{ful})}$ and $H_Z^{(\mathrm{ful})}$ are defined as $L/2 \times L$ block-circulant matrices, where each block row is obtained by a one-step cyclic shift.
  Example: with $J = 2$, $L = 6$

$$H_X^{(\mathrm{ful})} = \left(\begin{array}{ccc|ccc} F_0 & F_1 & F_2 & G_0 & G_1 & G_2 \\ F_2 & F_0 & F_1 & G_2 & G_0 & G_1 \\ \hline F_1 & F_2 & F_0 & G_1 & G_2 & G_0 \end{array}\right), \qquad H_Z^{(\mathrm{ful})} = \left(\begin{array}{ccc|ccc} G_0^{-1} & G_1^{-1} & G_2^{-1} & F_0^{-1} & F_1^{-1} & F_2^{-1} \\ G_2^{-1} & G_0^{-1} & G_1^{-1} & F_2^{-1} & F_0^{-1} & F_1^{-1} \\ \hline G_1^{-1} & G_2^{-1} & G_0^{-1} & F_1^{-1} & F_2^{-1} & F_0^{-1} \end{array}\right).$$

- The final parity-check matrices H_X and H_Z are obtained by taking the upper J block rows of $H_X^{(\mathrm{ful})}$ and $H_Z^{(\mathrm{ful})}$, respectively, resulting in (J, L)-regular LDPC matrices.

Generalized Hagiwara–Imai Construction: Commutativity Controls Orthogonality

Enforcing commutativity for all permutation blocks $\{F_i\}, \{G_j\}$ guarantees orthogonality, but it also induces low-weight logical errors and limits the minimum distance.

Key idea: relax commutativity.
Instead of enforcing commutativity for *all* permutation blocks, we require commutativity only for those block pairs that are strictly necessary to guarantee orthogonality between H_X and H_Z.

Commutativity controls orthogonality.
Orthogonality between H_X and H_Z is governed by the commutation relations among the permutation blocks F_i and G_j. Let

$$\Delta_J := \{(k-i) \bmod L_2 \mid 0 \le i, k \le J-1\} \subseteq \{0, 1, \ldots, L_2 - 1\}.$$

If, for every $r \in \Delta_J$ and every $u \in \{0, 1, \ldots, J-1\}$,

$$F_u \, G_{r-u} = G_{r-u} \, F_u,$$

then the orthogonality condition $H_X H_Z^{\mathsf{T}} = 0$ is satisfied.

Non-commutativity breaks distance limitations.
By deliberately allowing non-commutativity outside this necessary subset, the rigid algebraic structure is broken. As a result, stringent upper bounds inherent to fully commutative constructions, such as $d_{\min} \le L$ on the minimum distance and $g \le 2L$ on the girth, can be lifted.

Affine Permutation Matrices for Controlled Commutativity

- Commutativity among permutation blocks guarantees the orthogonality condition $H_X H_Z^{\mathsf{T}} = 0$, while deliberately introduced *non-commutativity* is a key ingredient for avoiding overly restrictive structures and lifting conventional upper bounds on the minimum distance.
- However, designing permutations that satisfy prescribed commutativity and non-commutativity patterns is highly nontrivial among general permutation matrices.

- **Affine permutation matrices (APMs).** An affine permutation on $\mathbb{Z}_P$ is

$$f(x) = ax + b \pmod{P}, \qquad a \in \mathbb{Z}_P^{\times},\ b \in \mathbb{Z}_P.$$

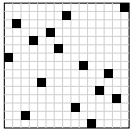
- For two affine permutations $f(x) = a_1 x + b_1$ and $g(x) = a_2 x + b_2$, the commutativity condition $f \circ g = g \circ f$ reduces to simple algebraic constraints:

$$(a_1 - 1)b_2 = (a_2 - 1)b_1 \pmod{P}.$$

- As a result, APMs allow commutativity and non-commutativity to be *explicitly designed* through algebraic parameters, rather than searched combinatorially.

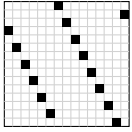
General permutation:

$f(x) = (6, 2, \ldots, 8, 11, 0)$

$F =$ 

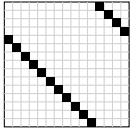
Affine permutation:

$f(x) = 2x + 3 \pmod{P = 15}$

$F =$ 

Circulant permutation:

$f(x) = x + 4 \pmod{P = 15}$

$F =$ 

Cycle Detection via Composition of APMs ($J = 2$)

$$H_X := \left(\begin{array}{ccc|ccc} F_0 & F_1 & F_2 & G_0 & G_1 & G_2 \\ F_2 & F_0 & F_1 & G_2 & G_0 & G_1 \end{array}\right), \qquad H_Z := \left(\begin{array}{ccc|ccc} G_0^{-1} & G_1^{-1} & G_2^{-1} & F_0^{-1} & F_1^{-1} & F_2^{-1} \\ G_2^{-1} & G_0^{-1} & G_1^{-1} & F_2^{-1} & F_0^{-1} & F_1^{-1} \end{array}\right) \quad (J = 2).$$

- Assume that all blocks F_i and G_i are **affine permutation matrices (APMs)**.
- The existence of a length-4 cycle in the corresponding 2×2 block submatrix is equivalent to the existence of a fixed point of a certain **composite affine permutation**, namely,

$$f_0^{-1} \circ f_1 \circ f_0^{-1} \circ f_2(x) = x \quad \text{for some } x \in [P].$$

- For APMs, the composite map is again affine,

$$f_0^{-1} \circ f_1 \circ f_0^{-1} \circ f_2(x) = ax + b \pmod{P},$$

  and the existence of fixed points can be checked by a simple **gcd condition**: $\gcd(a - 1, P) \mid b$.
- Therefore, the existence of 4-cycles is reduced to checking whether the corresponding **composition of APMs** admits a fixed point.
- This approach naturally generalizes to cycles of length ℓ. By exploiting the **ultra-sparse** structure induced by column weight $J = 2$, we can systematically avoid short cycles and achieve a girth of 12.

Non-Binary Extension

- We extend the parity-check matrices to the finite field $\mathrm{GF}(2^e)$. In this work, we use $e = 8$.
- Each finite-field element α^i is represented, in the parity-check matrix, by an $e \times e$ binary matrix A^i, where α is a *primitive element* defined by a primitive polynomial $p(\alpha) = 0$, and A is the corresponding *companion matrix*.
- Note that, since non-binary symbols are represented by binary companion matrices, the proposed code can be regarded as a *qubit code*, rather than a qudit code.
- A column weight of $J = 2$ is optimal, enabling **ultra-sparse** parity-check matrices and guaranteeing a **large girth (≥ 12)**.
- Belief propagation decoding updates probability distributions over $\mathrm{GF}(2^e)$.

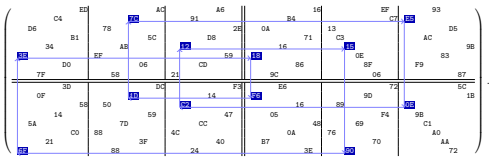


Figure: Example of non-binary parity-check matrix H_Z over $\mathrm{GF}(2^8)$.

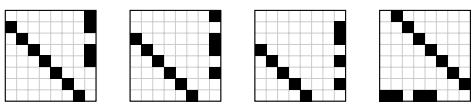


Figure: Companion matrix A and its powers $A^2, A^3, \ldots, A^{254} = A^{-1}$ over $\mathrm{GF}(2)$.

Eliminating Small Logical Errors for $J = 2$

- When the column weight is $J = 2$, every codeword is generated by *a single cycle or a union of cycles* in the Tanner graph.
- Such cycle-based structures in the *removed rows* $H_X^{(\mathrm{res})}$ and $H_Z^{(\mathrm{res})}$ naturally lead to *small-weight codewords*, which become dominant sources of logical errors.
- By **carefully assigning non-binary symbols** to the edges (or blocks), these cycles are made *rank-full*.
- As a result, the cycle-based structures *no longer form valid codewords*, effectively eliminating small logical errors.

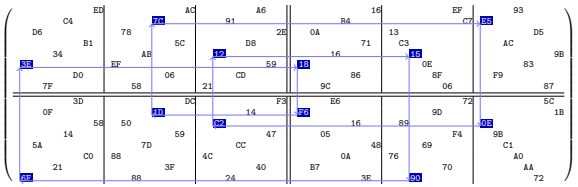


Figure: If a cycle of length 2ℓ is rank-deficient, then a cycle codeword of weight ℓ exists.

[2] Kasai, Hagiwara, Imai, and Sakaniwa, IEEE Trans. Information Theory, 2011.

Joint BP decoding[5]

- Non-binarized matrices: H_X and H_Z over $\mathrm{GF}(2^8)$
- Measure syndromes:

$$\boldsymbol{s} = H_Z \boldsymbol{x}, \quad \boldsymbol{t} = H_X \boldsymbol{z}$$

- Joint BP decoding exploits the sparsity of the parity-check matrix H and **iteratively updates the reliability** of each bit using local parity constraints.
- Joint BP iteratively and **simultaneously estimates** estimates $\hat{\boldsymbol{x}}$ and $\hat{\boldsymbol{z}}$ at each iteration.
- Error correction is regarded as successful if and only if

$$\boldsymbol{x} + \hat{\boldsymbol{x}} \in C_X^{\perp} \text{ and } \boldsymbol{z} + \hat{\boldsymbol{z}} \in C_Z^{\perp}.$$

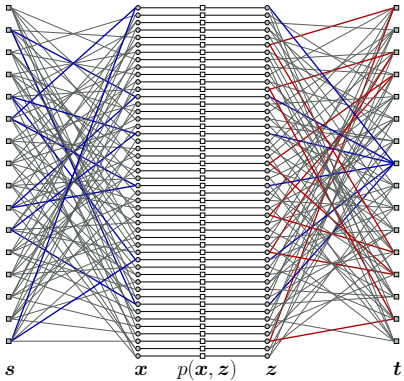
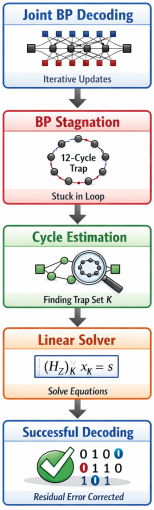


Figure: Factor graph of BP.

[5] https://github.com/kasaikenta/gd-css-decoder

Overview of Proposed Post-Processing Decoding Algorithm

1. First, we run joint belief propagation (BP) decoding for a sufficiently large number of iterations. In most cases, this step alone is enough to correctly estimate the noise.
2. In rare cases (approximately once in every 10^3 trials), joint BP decoding fails to converge. In such events, the number of unsatisfied syndromes does not decrease to zero and the decoding process stagnates, typically due to trapping in short cycles (e.g., cycles of length 12). When this happens, we proceed to estimate the corresponding trapping cycles.
3. Once the trapping cycles are identified, the remaining undetermined noise components are estimated by solving a small linear system of equations.



Method: Estimation of Trapping Cycles

- At each iteration, we keep track of the locations where the estimated noise and the syndrome values have recently changed:

  $K_d^{(\ell)}$: Estimated noise that have changed within the past d iterations

  $I_d^{(\ell)}$: Syndromes of the estimated noise that have changed within the past d iterations

- For sufficiently large ℓ and d, it was observed that $K_d^{(\ell)}$ and $I_d^{(\ell)}$ tend to **cover the columns and rows of trapping cycles**, respectively.
- This observation enables us to **efficiently identify the trapping cycles**.

	Estimation for x				Estimation for z			
ℓ	$\lvert K_{\mathrm{err}}^{(\ell)}\rvert$	$\lvert K_d^{(\ell)}\rvert$	$\lvert I_{\mathrm{err}}^{(\ell)}\rvert$	$\lvert I_d^{(\ell)}\rvert$	$\lvert K_{\mathrm{err}}^{(\ell)}\rvert$	$\lvert K_d^{(\ell)}\rvert$	$\lvert I_{\mathrm{err}}^{(\ell)}\rvert$	$\lvert I_d^{(\ell)}\rvert$
0	14944	0	9689	9689	15017	0	9741	9741
1	13731	4270	8618	10371	13845	4165	8677	10399
2	12875	6986	7676	10631	12959	6864	7791	10656
3	12108	8776	7036	10757	12306	8660	7178	10791
4	11693	10053	6558	10852	11765	10017	6717	10883
5	11297	11035	6221	10907	11370	11022	6304	10941
6	10866	11808	5862	10951	11043	11808	6028	10986
7	10542	12446	5640	10974	10667	12518	5745	11027
8	10300	12950	5464	10044	10364	13119	5537	10141
9	10069	11536	5216	9337	10099	11796	5334	9442
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
41	466	5682	462	3625	846	5956	684	3755
42	221	5088	204	3167	473	5405	436	3421
43	90	4337	103	2742	227	4822	225	3053
44	15	3633	25	2307	81	4243	95	2664
45	2	2980	2	1856	21	3575	27	2210
46	3	2257	4	1389	5	2882	7	1755
47	2	1595	2	909	0	2197	0	1300
48	2	973	2	565	0	1538	0	897
49	3	538	4	261	0	998	0	531
50	2	250	2	118	0	542	0	264
51	2	101	2	29	0	256	0	108
52	3	19	4	6	0	87	0	30
53	X stagnation region			6	0	23	0	7
54	2	6	2	6	Z converged region		0	0
55	3	6	4	6	0	0	0	0
56	2	6	2	6	0	0	0	0
57	2	6	2	6	0	0	0	0
58	3	6	4	6	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Figure: Transition of the joint BP decoding state over iterations ($d = 8$).

Method: Post-Processing Algorithm

- To estimate the X-noise, we **solve a linear system** involving the syndrome $\boldsymbol{s}$ and the noise vector $\boldsymbol{x}_K$ **associated with the trapping cycles**. Specifically, we use a submatrix of H_Z restricted to the set of column positions K corresponding to the trapping cycles:

$$\boldsymbol{s} = (H_Z)_K \boldsymbol{x}_K + (H_Z)_{\overline{K}} \hat{\boldsymbol{x}}_{\overline{K}}.$$

- The cardinality of K is independent of the code length. Hence, the system can be solved efficiently by Gaussian elimination with computational complexity $O(|K|^3)$.
- In our experiments, it was sufficient to take $|K| = 12$ to successfully resolve all trapping events.
- Similarly, the Z-noise vector is estimated by solving the corresponding linear system.
- Error correction is regarded as successful if and only if

$$\boldsymbol{x} + \hat{\boldsymbol{x}} \in C_X^{\perp} \quad \text{and} \quad \boldsymbol{z} + \hat{\boldsymbol{z}} \in C_Z^{\perp}.$$

Future Works and Conclusions

- We proposed a quantum LDPC coding framework inspired by non-binary LDPC codes, **achieving threshold-like** decoding performance close to the **hashing bound**.
- By combining ultra-sparse structures with column weight $J = 2$, affine permutation matrices, and carefully designed non-binary symbol assignments, we successfully eliminated **small logical errors**.
- The use of companion-matrix representations allows non-binary algebra to be exploited while keeping the physical code as a **qubit code**.
- As an important future direction toward fault-tolerant quantum computation (FTQC), it is crucial to incorporate realistic fault-tolerant noise models, including circuit-level noise, correlated errors, and **noisy and repeated syndrome measurements**.
- As an important future direction toward fault-tolerant quantum computation (FTQC), it is crucial to incorporate realistic fault-tolerant noise models, including circuit-level noise and correlated errors.
- We welcome collaborations on code constructions, decoding algorithms, noise modeling, and experimental validation toward practical FTQC.

(Optional): Conditions for Applying Density Evolution to Quantum LDPC Codes

- **Essential requirement for density evolution (DE):**
  - For a fixed number of BP iterations, the computation graph must be **locally tree-like**.
  - Under this condition, incoming BP messages can be treated as independent, which justifies the DE recursion.
- **Role of the girth:**
  - A girth that grows with the block length is a **sufficient condition** for guaranteeing local tree-likeness.
  - However, **girth growth is not a necessary condition**.
    - It is sufficient that, at any fixed depth, the neighborhood is tree-like with high probability.
    - In random sparse graph ensembles, local tree-likeness can hold even when the girth remains bounded.
- **Structural constraints specific to quantum LDPC codes:**
  - In the full $\mathrm{GF}(4)$ Tanner graph of stabilizer codes, commutativity constraints inevitably introduce **short cycles (in particular, 4-cycles)**.
  - In contrast, in the factor graph used in this work, cycles of length $2L$ arising from degenerate errors do exist, but they do not directly lead to decoding failures.
  - Moreover, no explicit upper bound on the girth is currently known, and we experimentally achieve large girth (e.g., 16).
- **Conclusion:**
  - Density evolution fundamentally relies on **local tree-likeness rather than on the girth itself**.
  - When the factor graph is appropriately designed, DE remains a meaningful analytical tool for quantum LDPC codes, even in the presence of quantum-specific constraints.