Lecture 9
27 Apr 2026
Ph/CS 219

Lifted Product Codes

We have established some tools for quantum code construction: Polynomial rings, chain complexes, BB codes, HGP codes, -- the next step is lifted product codes, which combine the HGP construction with polynomial rings.

The idea is to follow the HGP/chain complex strategy to satisfy the CSS conditions (commuting X and Z stabilizer generators) but where the matrix entries are elements of a ring rather than a finite field like $\mathbb{F}_2$.

We'll specifically consider the ring of univariant polynomials over $\mathbb{F}_2$ ($R = \mathbb{F}_2[x]/(x^L - 1)$). But this can be generalized to multivariate polynomials, group algebras of abelian groups (or even nonabelian groups; if we are careful to preserve the CSS conditions).

Recall the HGP construction

$$H_X = (A \otimes I_{n_B},\ I_{r_A} \otimes B^T) \qquad A = r_A \times n_A$$
$$H_Z = (I_{n_A} \otimes B,\ A^T \otimes I_{r_B}) \qquad B = r_B \times n_B$$

Register 1: $n_A n_B$ qubits

Register 2: $r_A r_B$ qubits

Number of X checks (rows of H_X): $r_A n_B$

Number of Z checks (rows of H_Z): $n_A r_B$

$$\Rightarrow n = n_A n_B + r_A r_B,\quad k \geq n - r_A n_B - n_A r_B = (n_A - r_A)(n_B - r_B)$$

Now, instead of the entries in A and B matrices being 0, 1, we choose them to be elements of the ring. In principle, these could be polynomials of any order up to L, but monomials are preferred, because we don't want the check weight to grow. Each entry is x^p where $p \in \{0, \_, L-1\}$, and we interpret x as a circulant matrix with a single 1 in each row — hence same for x^p

This construction is called a "lift" of the "seed matrices" A and B. In the seed code, entries are elements of R; in the lifted code, these elements of R are replaced by $L \times L$ matrices. These provide a compact and convenient way to describe very large check matrices. We may say that the lift turns the 2D structure of HGP construction into a 3D structure with a "fiber" (a cyclic 1D graph with L vertices) sitting above each vertex of a 2D "base". In this base, there are $n_A \times n_B$ sites in register 1 and $r_A \times r_B$ sites in register 2.

The lift increases the number of rows and columns in H_X and H_Z each by a factor of L, but without increasing check weight. For example, one row of A has n_B entries and each column of B has r_B entries. Each entry expands under the lift to an $L \times L$ matrix with weight 1 in each row. So the check weights are $n_A + r_B$ for H_X and $n_B + r_A$ for H_Z

27 Apr 2026 (3)
PHCS 219

The lifted code has

$$n = (n_A n_B + r_A r_B) L \quad \text{qubits,}$$

and the number of checks is

$$(r_A n_B + n_A r_B) L.$$

Hence we have $k \geq (n_A - r_A)(n_B - r_B) L$

(a lower bound because checks might not be independent).

A Z-operator that commutes with the X checks can be represented as a column vector (length n) over R, where each entry is a polynomial. The nonzero coefficients in that polynomial indicate, at a particular site of the base graph, on which of the qubits in the fiber the Z's act.

Therefore, each Z operator that commutes with X stabilizer corresponds to a solution to a set of polynomial equations of the form

$$v^T = (v_1, v_2 \cdots v_n) \quad X^{p_{i1}} v_{i_1} + \cdots + X^{p_{iw}} v_{i_w} = 0$$

- each term associated with a ~~[illegible]~~ nonzero entry of a row of H_X, where w is the weight of the row. The weight of the solution is the sum from 1 to n of the number of nonzero terms in the polynomial v_i.

Note that if v satisfies all the checks, then so does xv, x^2v, etc. Each solution lies on an orbit of L solutions. If

$$H_x v = 0 \quad \text{then} \quad x H_x v = 0 = H_x(xv)$$

Distributive property in the ring

Also, multiplication by x preserves the stabilizer, the image of H_Z^T. If v is in this image, then

$$v = H_Z^T u \text{ for some vector of polynomials } u$$

and $xv = xH_Z^T u = H_Z^T(xu)$ so xv in image

Therefore, each nontrivial Z-type logical operator lies on an orbit.

~~v in stab ⇒ v in stab~~

v commutes with X-stab $\Rightarrow$ so does xv

v not in stab $\Rightarrow$ neither is xv (or else $x^{-1}(xv) = v$ would be)

~~Something~~ To be avoided: In a given row of the seed with monomial entries

$$x^{p_1}, x^{p_2}, \ldots x^{p_n}$$

we don't want L to have common factors with all differences $p_2 - p_1$, $p_3 - p_2$, $p_4 - p_3$, etc. That could result in shorter orbits which might reduce the distance.

There is an upper bound on the distance d of this lifted product code, which arises from the $\mathbb{F}_2[x]$ variant of Cramer's rule in linear algebra. It says:

Consider a matrix which is $r \times (r+1)$. Because there are more columns than rows, its kernel is nontrivial. To construct an element of kernel, each component is the determinant of an $r \times r$ submatrix (same as a permanent over $\mathbb{F}_2$ because $+1 \equiv -1$). The permanent is a sum of $r!$ terms.

In the case of our $r \times n$ seed matrix, suppose $n > r$, and look at submatrix with $r+1$ columns. Each entry in the matrix is a monomial, so each term in the permanent is a product of monomials and hence a monomial. There are $r!$ terms, resulting in a polynomial with at most $r!$ terms. We construct a vector v with $r+1$ nonzero components annihilated by the seed matrix, where each component has weight at most $r!$, resulting in a total weight of $(r+1)!$.

In our lifted product construction with $n_A > r_A$ and $n_B > r_B$ this shows there are operators of weight no larger than $\min[(r_A+1)!, (r_B+1)!]$ but does not by itself give an upper bound on distance unless we can show this operator does not lie in the stabilizer. (see below.)

27 Apr 2026 (6)

An often-studied case is $A = B$:

$$H_X = (A \otimes I_n, I_r \otimes A^T) \quad \text{where } A \text{ is } r \times n$$
$$H_Z = (I_n \otimes A, A^T \otimes I_r)$$

Some examples are discussed in Camtel 2603.28627 (where it is claimed that $d \le (r+1)!$).

E.g. seed matrix 3×5 and $L = 33$

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & x^{14} & x^{29} & x^{11} & x^{26} \\ 1 & x^{13} & x^{2} & x^{15} & x^{21} \end{pmatrix}$$

Here $n = (5^2+3^2)33 = 34 \cdot 33 = 1122$

$k \ge (5-3)^2 33 = 132$ (actually 148)

$d \ge (3+1)! = 24$ (actually $d \ge 20$)

check weight is $3+5=8$

Another example: A is 3×7 and $L = 91$

$n = (7^2+3^2) \cdot 91 = 5278$

$k \ge (7-3)^2 \cdot 91 = 1456$ (actually 1480)

$d \le (3+1)! = 24$ (not known to be lower)

check weight $3+7=10$

Good rate $k/n = \dfrac{1480}{5278} = .28$

Increasing L maintains the high rate, but does not increase distance; we can't reach codes with $d = \Omega(n)$ with a fixed size q A

To get larger distance we can increase the size of the base, making it a double expander, but as we discussed that alone is not enough to break the $d = O(\sqrt{n})$ barrier, and just adding a fiber does not help — a more sophisticated LP construction with twisted fibers is this forces logical operators to spread, increasing their weight to $\Omega(n)$.

Back to the distance bound $d \leq (r+1)!$. We find a vector v of weight $(r+1)!$ in the kernel of A, $Av = 0$. So there is a vector $v \otimes e$ supported on first register in kernel of H_X. This can't be in stabilizer because $H_Z = I \otimes B$ on the first register.

Another way to say this, we constructed a representative of $H_1(A)$ otimes $H_0(B^T)$. $H_1(A)$ is just the code with A parity check, and $H_0(B^T)$ is the space of vectors modulo those generated by rows of B^T (dual to code defined by B). If B is $r \times n$ with $r < n$, there exists a weight-1 vector not in $\mathrm{Im}(B^T)$.

The nice structure of our LP codes has benefits for syndrome extraction, syndrome decoding, logical operations. Logical ops are a particular worry in high-rate codes as there are many "overlapping" codewords distributed among the same physical qubits.

we'll eventually discuss a solution: "teleporting" from a memory code to a processor code via ancilla-assisted measurements.

Let's discuss syndrome measurement, first for HGP code with $H_X = (A \otimes I_n, I_r \otimes A^T)$

$H_Z = (I_n \otimes A, A^T \otimes I_r)$

Arrange 2 registers in $n \times n$ and $r \times r$ arrays

$A \otimes I_n$ acts nontrivially on qubits in same row of Register 1

$I_r \otimes A^T$ " " " column of Register 2

Suppose data is fixed and ancillas are mobile. For each row in A, ancilla interacts in each row of array with qubit positions where that row has support. We flash the Rydberg laser as ancilla pass data qubit where A has support. Do this in all rows of the arrays in parallel.

We're not done. Each ancilla qubit after passing by n data qubits in a row of register 1 must then pass through r qubits in a column of register 2. For now, there is a bottleneck: we need more clever procedures for scheduling and routing.

Discussed by Xu et al 2308.08648. Two ideas help. ① Color scheduling. Color edges of Tanner graph so each check & variable connects to unlike colors. # of colors = degree of the bipartite graph. In a timestep perform interactions of one color. Each ancilla+data qubit interacts with one other qubit. ② "Divide & conquer." Minimize the number of moves with a recursive routing algorithm.