Quantum Error Correction Codes

Dong-Sheng Wang

April 21, 2021

Definition. *A quantum code is usually defined by a linear encoding isometry* $V : \mathcal{H}_l \to \mathcal{C} \subset \mathcal{H}$*, and an error correction code is a triple* $(V, \mathcal{N}, \mathcal{R})$*, which requires the ability to fully recovery any state* $|\psi\rangle \in \mathcal{C}$ *by a recovery channel* $\mathcal{R}$ *after a prescribed noise channel* $\mathcal{N}$ *acting on the code space* $\mathcal{C}$*.*

Keywords

Encoding; linear codes; error correction; approximate codes; coding bounds; stabilizer codes; toric code; LDPC codes; fault tolerance; universality; threshold; error control; self correction.

Abstract

We explain quantum codes and fault-tolerant quantum computing. In order to perform reliable quantum computing, we have to use qubits encoded in error-correction codes. We first explain the basics of quantum codes, including encoding isometry, error correction and detection, code distance, encoding bounds, logical gates, recovery channel, etc. Then we introduce stabilizer codes and describe some examples. Next we explain how to use quantum error-correction codes for quantum computing,

with a notable candidate: color code. Finally, we survey other ways of dealing with decoherence and some frontiers, and also stories.

1 Minimal version

1.1 Opening

The subject of quantum error correction (QEC) has been developed for two decades, just a little bit shorter than quantum computing itself. QEC is the most crucial part for quantum computing since it ensures the computation to be reliable, otherwise the output of a computing device is just crap. So every quantum computer scientist has to understand QEC and the framework of fault-tolerant quantum computing.

The very starting idea is *encoding*, which also lies at the heart of all classical communication, computing, cryptography, and relevant fields. The encoding can be used to protect against noise or enemy. Encoding refers to the process to use redundancy to enhance the robustness of information against noises (errors). For instance, we can encode 0 as a string of 0s and 1 as a string of 1s, so that flips of a few bits do not affect the information we encode. Information processing occurs in an encoded way: first *encode*, then do *operations* you want, then *decode*, and finally *readout* the results you need. Furthermore, encoding also occurs in natural physical systems: macroscopic observable is encoded in microscopic details of statistical systems, the interior bulk property of an object can be encoded in its boundary, etc. A good encoding often connects with appealing physics, and finding good codes of course also requires skillful work.

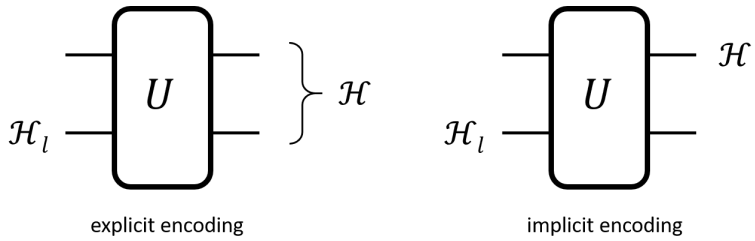

Figure 1: The explicit encoding vs. implicit encoding. An encoding isometry V is often embedded in a unitary U.

1.2 Basics

Let's first explain what an encoding is. For quantum codes, we often use linear encoding defined by an isometry $V : \mathcal{H}_l \to \mathcal{C} \subset \mathcal{H}$ which satisfies

$$V^\dagger V = \mathbb{1},\ VV^\dagger = P, \tag{1}$$

for P as the projector on the code space $\mathcal{C} \subset \mathcal{H}$. Also we require $\dim(\mathcal{H}) > \dim(\mathcal{H}_l)$ so that there are redundancy to encode states.

Which one is more fundamental, V or P for a code? It turns out both are okay. A code defined by V is *explicit* since it shows the map $|i\rangle \mapsto |\psi_i\rangle$ with $V = \sum_i |\psi_i\rangle\langle i|$ for an orthonormal basis $\{|i\rangle\}$ of $\mathcal{H}_l$, and a code defined by P is *implicit* since the map $|i\rangle \mapsto |\psi_i\rangle$ is only described classically, and it still needs to construct V separately. For instance, the code $|0\rangle \mapsto |000\rangle$, $|1\rangle \mapsto |111\rangle$ which encodes a qubit into three qubits is explicitly defined by its encoding circuit. The code $|0\rangle \mapsto |n = 0\rangle$, $|1\rangle \mapsto |n = 1\rangle$ which encodes a qubit into number states of an oscillator is implicit since the encoding circuit is often not specified. See figure 1 for their differences. In general, for explicit encodings the physical space $\mathcal{H}$ contains the original logical space $\mathcal{H}_l$, and for implicit ones it is not the case.

Now we shall discuss noise operators, or called error operators, which can disturb information in $\mathcal{C}$ by causing leakage to $\mathcal{C}^\perp = \mathcal{H} \ominus \mathcal{C}$ or making changes in $\mathcal{C}$ itself. Do we have to correct the noises? We don't have to if we do not need the information to be fully protected, depending on the tasks. For instance, the noises may be weak for some communication channels, so codes can be designed just to protect against particular types of noises so that the fidelity of output can be high enough. In other words, we do nothing for recovery. However, for error correction we need to apply a recovery channel, $\mathcal{R}$, and also for error detection we need to apply the projector P onto the code space.

For an error operator A, the simple correction scheme (i.e. recovery) is to apply its inverse A^{-1}. In quantum theory, we often consider unitary operators or quantum channels defined by a set of Kraus operators. One central result of QEC is that given a code $\mathcal{C}$ and a set of error operators $\{E_i\}$, which shall form a channel $\mathcal{N}$ with condition $\sum_i E_i^\dagger E_i = \mathbb{1}$, the errors can be corrected iff

$$PE_i^\dagger E_j P = c_{ij} P \tag{2}$$

for $[c_{ij}]$ as a non-negative matrix, known as *coding state* $\rho_{\mathcal{C}}$. The recovery is not by E_i but linear combination of them. Namely, the coding state $\rho_{\mathcal{C}}$ can be diagonalized with eigenvalues d_a, and the condition will become

$$PF_a^\dagger F_b P = d_a \delta_{ab} P. \tag{3}$$

The recovery channel is defined by a set of operators $R_a = \frac{1}{\sqrt{d_a}} F_a^\dagger$. It can be easily checked that

$$\mathcal{R}\mathcal{N}(\rho) = \rho, \ \rho \in \mathcal{C}. \tag{4}$$

The condition above means that different F_a acting on P leads to orthogonal spaces so they can be distinguished.

Quantum error detection (QED) can be viewed as a simple version of QEC. The condition for QED of a channel is

$$PE_iP = e_iP, \tag{5}$$

which is a weaker condition than QEC. This means that a code can detect more errors than correct them.

A crucial property of QEC is that if a code corrects a set $\{E_i\}$, then it also corrects any operators in the linear span of $\{E_i\}$ by the same recovery $\mathcal{R}$. So, strictly speaking, the correction is not about a single channel; instead, it is about an operator space, which covers many more channels. This "linear-span" property has profound consequences. For instance, usually we consider codes using many qubits, and we only need to consider errors on each qubit, and the spanning qubit errors are just the Pauli operators: X, Y, Z. Correcting Pauli errors will guarantee the correction of other error operators. As such, we often speak of "number of errors", which refer to the weight of arbitrary errors of the form $\otimes_n A_n$ for n as labels of qubits.

The opposite of noise operators are *logical* operators, also known as *gates*. Here we only consider unitary ones. A unitary U acting on $\mathcal{H}$ is logical iff

$$[U, P] = 0. \tag{6}$$

That is, it preserves the code space. The set of logical operators forms a group, usually $SU(d)$, with the code space $\mathcal{C}$ as a representation of it. There are also hybrid operators which are both logical and noisy: they lead to logical operations yet with leakages. Hybrid operators are dangerous for a code since they

are not correctable, and they can determine the actual *distance* of a code.

The distance of a code d_c is defined as the minimal "size" of operators that can lead to a logical operator. The notion of distance refers to distances among codewords $|\psi\rangle \in \mathcal{C}$. Higher distance means the code is more robust against noises. Here "size" can be measured by various quantities depending on the contexts. There are two basic settings: when $\mathcal{H}$ is of the form $\otimes_n \mathcal{H}_n$ or the form $\oplus_n \mathcal{H}_n$. We often consider weight of operators with tensor-product forms, and power of operators for the other case. For instance, the weight of a tensor-product of Pauli operators $\otimes_n \sigma_n$ is the number of non-identity σs in it. Now given a distance d_c, it is clear to see (from the QEC condition) it means $(d_c - 1)$ errors can be detected, while $\frac{(d_c-1)}{2}$ errors can be corrected.

Let's focus on codes with multiple qubits which are the common setting. A code of n qubits encoding k logical qubits with distance d is denoted as $[[n, k, d]]$, the double-bracket meaning "quantum" but here we use the simple notation $[n, k, d]$. A natural question is: are their any relations among n, k, d? It turns out there are some tradeoffs from coding bounds, and the well-known ones are Hamming bound, Singleton bound, and Gilbert-Varshamov (GV) bound. The Singleton bound says that

$$n - k \geq 2(d - 1), \tag{7}$$

while the classical version of it is $n-k \geq (d-1)$. The difference comes from the fact: for qubits we need to correct both X and Z, but for classical bits we only need to correct Z. To be precise, let's see how to prove it. Consider an orthonormal basis $\{|i\rangle\}$ for k qubits and the Bell state $|\Phi\rangle = \frac{1}{\sqrt{2^k}} \sum_i |ii\rangle$ of $2k$ qubits,

and then apply the encoding isometry V to half of them. We now have $|\Phi'\rangle = \frac{1}{\sqrt{2^k}} \sum_i |i, \phi_i\rangle$ for $k+n$ qubits. Divide the n qubits to groups of $d-1$, $d-1$, and $n-2(d-1)$ qubits, with label A, B, and C. Label the other k qubits as R. Now the code distance d means $\rho_{RA} = \rho_R \otimes \rho_A$, $\rho_{RB} = \rho_R \otimes \rho_B$. Using entropy relations $S_{AC} \leq S_A + S_C$, $S_{BC} \leq S_B + S_C$, we will get $k = S_R \leq S_C \leq n - 2(d-1)$. In other form, it is

$$\frac{k}{n} \leq 1 - 2\frac{(d-1)}{n}. \tag{8}$$

The Hamming bound and GV bound arise from a counting of distinct errors, which is, in order, an upper bound

$$\frac{k}{n} \leq 1 - (\frac{d}{2n}) \log 3 - h(\frac{d}{2n}), \tag{9}$$

and a lower bound

$$\frac{k}{n} \geq 1 - (\frac{d}{n}) \log 3 - h(\frac{d}{n}), \tag{10}$$

with $h(x)$ the binary entropy of x. The Hamming upper bound is more tight than the Singleton upper bound, but the later also applies to *degenerate* code, which means some errors are equivalent and can be corrected by the same recovery scheme. We will show below that topological codes are highly degenerate, and this greatly benefits the design of good codes.

1.3 More: stabilizer codes

For multi-qubit codes, a large class of codes is the stabilizer codes, for which a code projector P is defined from a set of so-called *stabilizers* $\{S_i\}$, which are low-weight product of Pauli

operators and commuting. There is a nice group-theoretic description: the stabilizers form an Abelian group, the stabilizer group, and for a code $[n, k, d]$, there are $(n - k)$ stabilizer generators. The errors to consider are also low-weight product of Pauli operators, and correctable errors anti-commute with some stabilizers. The anti-commutation pattern forms the syndrome, and for non-degenerate stabilizer codes each error has distinct syndrome. A signature of non-degenerate stabilizer codes is that the weight of stabilizers have to be larger than d since otherwise the product of two errors $E_a E_b$ can be a stabilizer, then E_a and E_b cannot be distinguished (see the QEC condition). In this section, we will use examples to reveal features of stabilizer codes.

From Singleton bound, the minimal n to encode a qubit with distance 3 is 5. This is the so-called five-qubit code, or XZZX code since the stabilizers are $XZZX\mathbb{1}$, $\mathbb{1}XZZX$, $X\mathbb{1}XZZ$, $ZX\mathbb{1}XZ$, $ZZX\mathbb{1}X$. It is non-degenerate. Each stabilizer can be split as a product, e.g., $XZZX\mathbb{1} = (XYX\mathbb{1}\mathbb{1})(\mathbb{1}XYX\mathbb{1})$, with $XYX\mathbb{1}\mathbb{1}$ as a logical operator. By passing, this is actually from the cluster state. The code is implicit since we do not know the encoding circuit. To prepare the code, we can prepare a five-qubit cluster state by expressing it as a matrix-product state with periodic boundary condition. We can also use $XXXXX$ as a logical operator which anti-commutes with $XYX\mathbb{1}\mathbb{1}$.

More interesting codes will involve more qubits. We do not intend to review all known stabilizer codes since there are too many. Below we survey some types of them, and these types may overlap.

- CSS codes: all stabilizers are product of X or product of Z. A CSS code can be constructed from two classical codes. An example is Shor's 9-qubit code, with stabilizers of the

form ZZ and $\otimes_n X_n$, similar with the Hamiltonian of Ising model.

- topological (or local) codes: the physical support of $\mathcal{H}$ is on a regular manifold and the support of each stabilizer is geometrically local and a constant. Logical gates are topological. Examples are toric code and color code, which physically are spin liquids.

- subsystem codes: the code space $\mathcal{C}$ is defined by a set of non-commuting local gauge operators, while a set of semi-local stabilizers also exist. The space has two parts $\mathcal{C} = \mathcal{T} \otimes \mathcal{G}$, with $\mathcal{T}$ as the true code space and $\mathcal{G}$ as a gauge space that does not encode information but participate the error correction. An example is Bacon-Shor code, also physically known as the quantum compass model.

- finite-rate LDPC codes: the encoding rate $r = \frac{k}{n}$ stays finite as $n \to \infty$, and LDPC means low-density parity check, namely, each qubit is involved in a constant number of stabilizers, and each stabilizer involves a constant number of qubits (but may not be local). An example is the hypergraph-product codes.

The topological codes are famous since they are robust against many kinds of local noises. However, the redundancy is huge and this is revealed from the bound

$$kd^2 \leq O(n), \tag{11}$$

on 2D codes. It means the encoding rate r goes to zero if the distance d increases with n. It's beneficial to see how this is proved. First, it divides the system into three parts: correctable

parts A and B, with the same size, and their boundary C that is not correctable. This is similar with the proof of Singleton bound, and it shows $k \leq \frac{n}{R^2}$ for R as the linear size of A and B. Next it shows that $R \sim d$ which is not hard to see. The proof is consistent with area law of entanglement which says that the entropy of a region scales with its boundary size, namely, information of a bulk is encoded in its boundary.

Logical gates on topological stabilizer codes are also constrained. First recall that Pauli operators X, Y, Z with $\pm i\mathbb{1}$ form a group, called Pauli group, $\mathcal{P}$, which is a projective representation of the Klein-four group $Z_2 \times Z_2$. This extends to n-qubit case as $\mathcal{P}_n$. Now the so-called Clifford hierarchy is defined as a set of sets, and each set with integer label $k > 1$, the 'level', is defined as

$$\mathcal{C}_k = \{U | UPU^\dagger \in \mathcal{C}_{k-1}, \forall P \in \mathcal{P}_n \equiv \mathcal{C}_1\}. \tag{12}$$

The $\mathcal{C}_2$ is known as Clifford group, which contains Hadamard gate H, phase gate S, CNOT, and their products. Consider logical gates from finite-depth local unitary (FDLU) operators, which do not change the topology of the code. It is shown that D spatial topological stabilizer codes can support FDLU logical gates from Clifford hierarchy for levels at most D. The proof is very interesting and it uses a partition of the system into $D+1$ correctable regions, Λ_j, different from the partition for the proof of encoding rate bound above. See figure 2 for a comparison. Consider a collection of D different Pauli logical operators P_j, and require the support of P_j does not overlap with Λ_j. This is always possible. The idea is to consider the commutator

$$K_j = P_j^\dagger K_{j-1} P_j K_{j-1}^\dagger \tag{13}$$

for $K_1 = UP_1U^\dagger$ with any FDLU U. Different K_j has different

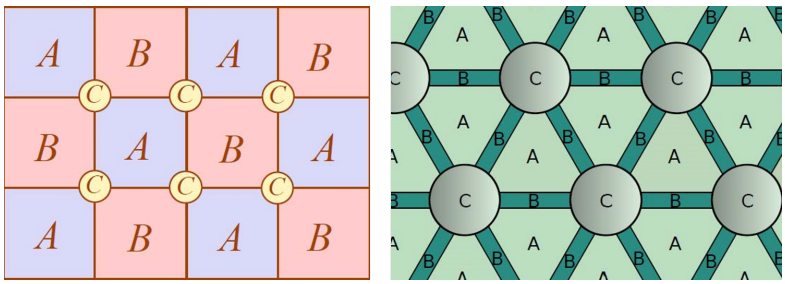

Figure 2: The partition for a topological stabilizer code for the proof of encoding rate (left) and logical gates (right).

supports, and it shows the support of K_D is the last one: Λ_{D+1}. So logically $K_D = \pm\mathbb{1}$ and it is easy to show $U \in \mathcal{C}_D$.

Physically, it is not easy to understand the meaning of K_j. But we can treat gates in $\mathcal{C}_D$ and lower as a set of *order parameters* that define the topological order of the code since FDLU operations shall not change the topological order. For instance, for the toric code the FDLU logical gates include: Pauli gates which are loops, H gate and CNOT which are depth-one global operations, S gate which is slightly more complicated. The Pauli loops define the topological order of toric code, which has ground state degeneracy as four robust against local perturbations. The H gate, which maps between X and Z, is a duality of the code that preserves the topological order. The same holds for S gate which maps between X and Y. The CNOT gate couples two toric codes together while still preserves the topological order for each of them. The above result shows that higher-dimensional topological order needs more order parameters to characterize it, not just the Pauli operators.

2 Advanced topics: fault-tolerant quantum computing

Now let's move on to discuss how to use quantum codes for quantum computing. This is the framework of fault-tolerant quantum computing (FTQC), believed to be the inevitable model of reliable quantum computing. We will analyze the notion of *universality*, what codes to use, the QEC algorithm, how to perform gates, and how to carry out computing tasks.

A computing process of FTQC looks like this

$$G_1 \cdot \text{QEC} \cdot G_2 \cdot \text{QEC} \cdots \tag{14}$$

for G as various gates. We omit the initial state and final readout. The first issue is about universality, namely, what shall we achieve in FTQC. Universality means that any gate $U \in SU(2^n)$ for any n can be realized efficiently to arbitrary accuracy ϵ. The efficiency means the cost of quantum computing, mainly the number of primary gates, scales as a polynomial of $\log \frac{1}{\epsilon}$ and n. The accuracy ϵ can be quantified by operator norm $\|U - U'\|$ or its equivalence which does not depend on initial states. The key point to note is that the accuracy can be arbitrarily small, otherwise it is not universal.

To approximate U, which contains continuous rotation angles, we shall not use continuous-variable gates; instead, we use a few gates with fixed form, and these gates are "digital". Namely, there exists universal gate set so that any U can be approximated well by a sequence of gates from a universal set. To prove the universality of a gate set is nontrivial, but here we only list a few examples: the set $\{H, T, CZ\}$, $\{H, CCZ\}$, $\{H, CS\}$, for CZ as controlled-Z, CCZ as controlled-CZ, CS as controlled-S, with S as phase gate. Given $U \in SU(2^n)$, finding a sequence form

of U' is a classical optimization problem that has been widely studied. At the end, all gates in FTQC shall be these primary gates from universal gate sets.

Note the gates above are logical and we still need to figure out how to realize them physically. Recall that the logical part of U is PUP. The simplest form of gate is the so-called *transversal* gates

$$U = U_1 \otimes U_2 \otimes U_3 \cdots \tag{15}$$

with each U_n sandwiched by QECs on the site n. However, there is a no-go theorem says transversal gates cannot be universal; probably because it is too good. An extension of it is the FDLU which was mentioned above. These gates do not spread out errors too far due to the finite depth so do not jeopardize QEC. However, finding FDLU logical gates is difficult given a certain code.

At this point we shall mention the anyonic quantum computing, which realizes logical gates by braidings of non-Abelian anyons. Braidings are linear-depth local unitary circuits, so they can spread out local errors to be nonlocal ones. QEC needs to be done after each braiding to ensure fault tolerance, and this is not an easy task in practice. We will review anyonic quantum computing separately.

Now for codes and QEC, which codes to choose for FTQC? There shouldn't be a unique answer since different codes have different merits. We can just use a single code if it is powerful enough, but so far no such code is known. So we have to use combinations of codes. Some primary ways are as follows:

- Concatenation: use encodings in sequence. e.g., Shor's code.

- Switching: use encodings alternatively. For stabilizer codes, this means to use stabilizer measurements to convert between codes.
- Augmentation such as state injection: use additional resources beyond the codes. e.g., magic-state injection for the T gate which can be used for toric code FTQC.

The errors encountered in different schemes are different, so the QEC are also different. However, a common thing is the physical error rate r_p has to be below a *threshold*, r_p^*, so that the logical error rate r_l can be made arbitrarily small by increasing the distance of the code. This is the content of the *threshold theorem*. Just like the classical repetition code, the noise on each bit shall not be too strong in order for the code to be reliable.

The QEC procedure becomes tedious for large codes such as topological codes. A *decoder* is a classical algorithm which is necessary to decide the recovery scheme from the syndrome. For degenerate codes, there could be various distinct decoders and finding optimal decoders is nontrivial. Decoders are often used in numerical studies.

A well-studied setup for FTQC is based on (gauge) color code. We will not discuss the details here but the scheme is quite simple. Color codes are a family of 3D CSS codes and there is a color code with transversal CZ and T gates, but not H gate. The idea is to use code switching to a gauge color code which has transversal H gate. However, the codes are 3D and the weights of stabilizers are not so small. Compared with toric code, no magic state is needed which is a great advantage of color codes.

Finally, we have to know how to perform a computing task. For short, quantum computing is just a sequence of gates, namely,

a quantum circuit. However, we need to know how a circuit is designed and how it performs. This often requires some classical algorithms and this leads to the classical-quantum *hybrid* paradigm. There are two types: a) loop-free: there is a classical algorithm $\mathcal{A}$ that design a quantum circuit $\mathcal{Q}$. b) feed forward: there is a classical algorithm $\mathcal{A}$ that can change a quantum circuit $\mathcal{Q}$ depending on the measurement output $\mathcal{M}$ from $\mathcal{Q}$. Quantum machine learning is such an example. In addition, we can also envision a fully quantum paradigm, but this is not well understood yet.

3 Relevant theory: dynamical decoupling, error mitigation, randomized benchmarking

A universal FUQC is very powerful, and there is no hope to build one within decades. Just like the classical case, there could be various types of non-universal computing devices that are also useful, such as simulators, detectors, sensors, calibrators, etc. These devices do not require powerful QEC but still can have some ability to fight against decoherence or noises. Here we sketch some approaches.

We can try to suppress errors instead of correcting them. A technique is known as dynamical decoupling (DD), which uses rapid time-dependent controls and relies on Magnus expansion of time-dependent evolution. The external noises is expected to average out when the control rate is faster than the time scale of noisy processes. The noisy process $\mathcal{L}$ together with the DD scheme $\mathcal{D}$ shall be close with a unitary process U. For quantum computing, DD is used to extend the lifetime of qubits while compatible with execution of gates.

Instead of dealing with evolution, the error mitigation (EM) technique focuses on observable. Expressing the ideal state as a linear combination of noisy ones, the ideal observable can be approached by taking linear combination of noisy ones. This requires to run the noisy circuit many times, each with different noise configurations. The number of samples is argued to be efficient with the desired accuracy of observable.

Errors do not only arise from external noises. Imperfect physical operations to realize gates also introduce in errors. In QEC we often assume this kind of control errors can be treated as external errors, but this is not the case for other settings. A technique to calibrate and improve gates is known as randomized benchmarking (RB). The basic idea is to use random circuits composed with random gates that would do nothing if the gates were perfect, and then measure observable. From these samples it can extract the average gate infidelity. Furthermore, the gate operations may depend on a few parameters, and an optimization is possible based on RB to increase the gate fidelity by tuning the parameters.

4 Frontiers

There are lots of frontiers, although the first-generation quantum computers (or simulators) are already in use, e.g., moderate-scale optical lattices, superconducting qubit networks, linear optics networks, etc. These simulators are not fault tolerant.

A frontier is to find better LDPC codes, as hinted by the coding bounds. As we know, CSS codes are combinations of classical codes. So it is natural to consider combinations of quantum codes. Some recent examples are homological prod-

uct codes, hypergraph product codes, fiber bundle codes, etc. The goal is to have both the encoding rate $\frac{k}{n}$ and distance d increasing with the system size n. These codes are mainly constructed mathematically, and it is not clear yet what physical phases of matter they correspond to (probably gapless phases with long-range interactions). Also FTQC with them just begin to develop.

A subject we did not discuss is approximate codes which do not fully recovery logical states. This paradigm lies in between the FTQC and noisy computing. There are some approximate codes such as GKP code, VBS codes, holographic codes, etc, but a full theory of quantum computing with approximate codes is not developed yet. These codes do not obey the standard coding bounds, but on the other hand, the accuracy of such computing is limited.

Anyonic quantum computing is a setup people are trying to realize in labs, and there is a hope to achieve this via Majorana systems in a decade. Although some non-braiding operations are needed to achieve universality, this system could be better than others such as superconducting computers.

A theoretical subject is about self-correcting codes. This is motivated by the classical example: 2D Ising model. We can encode a classical bit in the magnetization, which is robust against thermal noises below the critical temperature T_c. The encoding is a subsystem code, since we only need the sign of magnetization instead of its values. The 4D toric code is a self-correcting quantum code, which is physically a spin liquid, but clearly we cannot realize it easily in 3D. There is a hope to find new ones in fracton orders or finite-rate LDPC codes, which physically rely on non-thermal, glassy, or other mechanism instead of gas

or liquid mechanism.

5 History, people, and story

As we mentioned, the subject of QEC starts just a bit late than quantum computing itself. The stabilizer framework was established in 1997 and becomes the main stream of quantum codes. Probably lots of computer or information scientists think they are the only kind of quantum codes. Stabilizer states are very 'classical', in the sense that they can be efficiently represented classically, and operations that only change stabilizers to stabilizers can also be efficiently simulated on classical computers. The errors to deal with are Pauli X and Z, which is just two 'copies' of errors instead of one for classical codes. But superposition of stabilizer states are not stabilizer states anymore, and this is the origin of being quantum. Physicists are also developing non-stabilizer codes which are not so intuitive for computer scientists. Most likely, they will deviate from each other further away in the near future, dividing the world of quantum codes into stabilizer and non-stabilizer parts.

The features of topological stabilizer codes were mainly studied by S. Bravyi, a leading theorist at IBM, around 2010. The results are, however, mostly negative regarding encoding rate, logical gates, self correction, etc. However, this motivates people to look for better codes with topological codes as an ingredient or mirror. He also works on many other subjects such as quantum algorithms, computational complexity, anyonic quantum computing, etc all with high-impact results. It is fair to say he is the kind of genius who is required to be there to clean up the way in the field of quantum computing, as commonly to be the

case for any field.

Another phenomena to notice is there are not many Chinese in the subject of QEC, given the large amount of researchers on quantum computing. This situation will change gradually since quantum simulators will prove to be highly limited, and FTQC seems to be the right choice in the long run.

References

- Daniel Gottesman, An Introduction to Quantum Error Correction and Fault Tolerant Quantum Computation, arXiv:0904.2557.
- Barbara Terhal, Quantum Error Correction for Quantum Memories, Rev. Mod. Phys. 87, 307 (2015).

Concept map