Efficient decoding of random errors for quantum expander codes

Omar Fawzi & **Antoine Grospellier** & Anthony Leverrier

February 13, 2018

Main motivation: fault-tolerant quantum computation

Threshold Theorem [Ben-Or & Aharonov, '97]

We can simulate a quantum circuit with T perfect gates and m logical qubits by a fault-tolerant circuit with noisy gates and $\mathcal{O}(m \operatorname{polylog}(mT))$ physical qubits.

Main motivation: fault-tolerant quantum computation

Threshold Theorem [Ben-Or & Aharonov, '97]

We can simulate a quantum circuit with T perfect gates and m logical qubits by a fault-tolerant circuit with noisy gates and $\mathcal{O}(m \operatorname{polylog}(mT))$ physical qubits.

- Practice: break RSA with 4000 logical qubits, but $10^6 - 10^9$ physical qubits
- [Gottesman, '13] improved this result using constant rate quantum codes instead of concatenated codes

Threshold theorem with constant overhead [Gottesman, '13]

Provided codes with nice properties exist, the ratio physical/logical qubits can be made constant: $\mathcal{O}(m \operatorname{polylog}(mT)) \rightsquigarrow \mathcal{O}(m)$

Main motivation: fault-tolerant quantum computation

Threshold Theorem [Ben-Or & Aharonov, '97]

We can simulate a quantum circuit with T perfect gates and m logical qubits by a fault-tolerant circuit with noisy gates and $\mathcal{O}(m \operatorname{polylog}(mT))$ physical qubits.

- Practice: break RSA with 4000 logical qubits, but $10^6 - 10^9$ physical qubits
- [Gottesman, '13] improved this result using constant rate quantum codes instead of concatenated codes

Threshold theorem with constant overhead [Gottesman, '13]

Provided codes with nice properties exist, the ratio physical/logical qubits can be made constant: $\mathcal{O}(m \operatorname{polylog}(mT)) \rightsquigarrow \mathcal{O}(m)$

- Before this work, no existing codes had these "nice properties"
- We proved that quantum expander codes have these "nice properties"

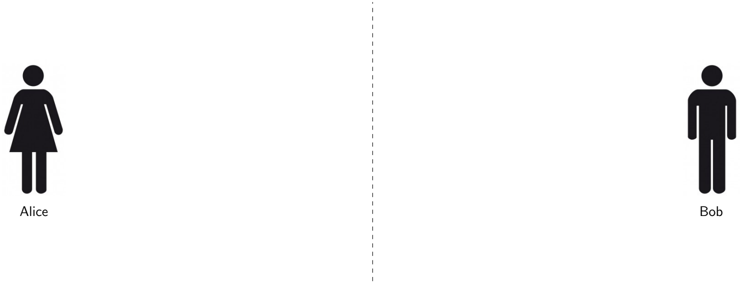
Content of the talk

1. Classical error correction
2. Quantum error correction
3. Our contribution

Outline

1. Classical error correction
2. Quantum error correction
3. Our contribution

Classical error correction



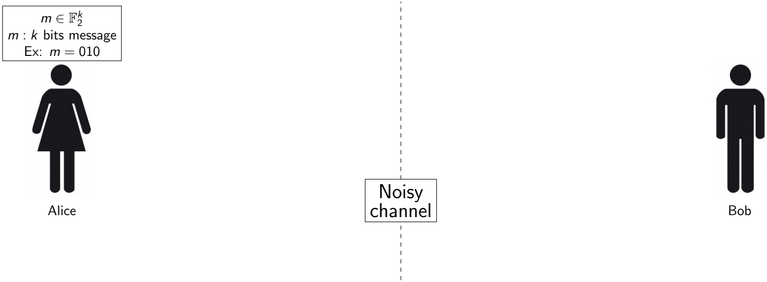
Classical error correction

$m \in \mathbb{F}_2^k$
m : k bits message
Ex: $m = 010$

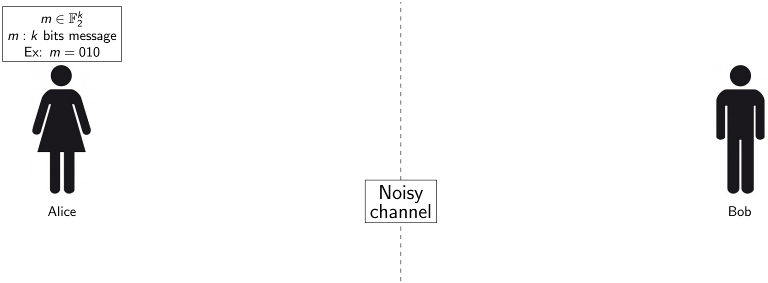
Alice

Bob

Classical error correction

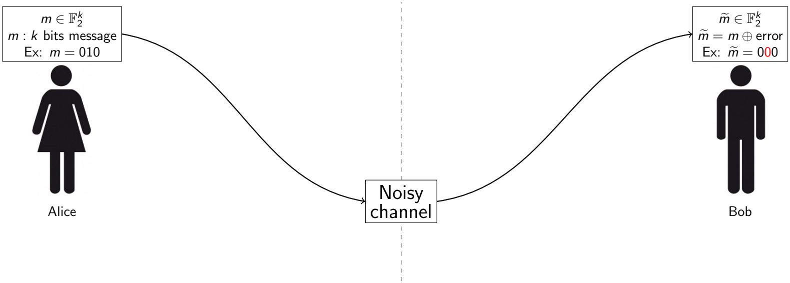


Classical error correction



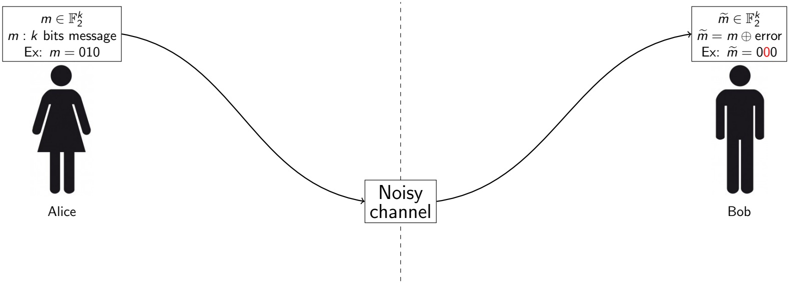
Without error correcting codes

Classical error correction



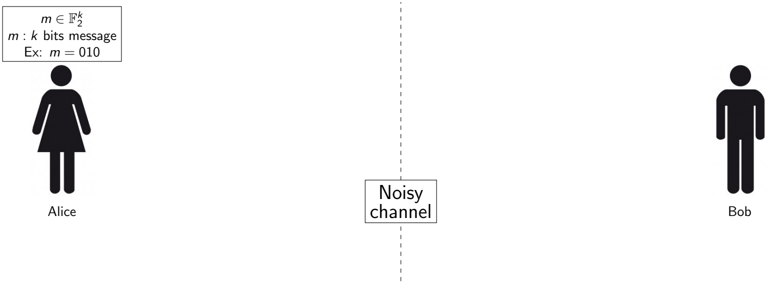
Without error correcting codes

Classical error correction



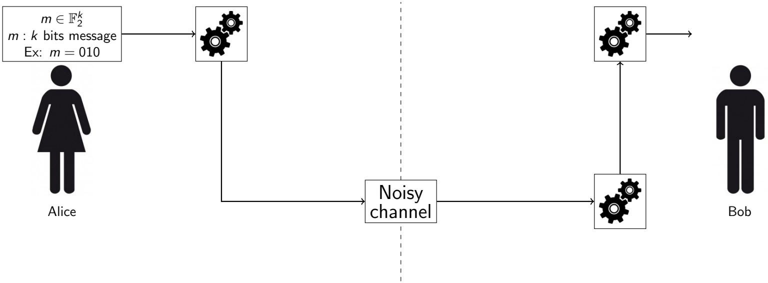
Without error correcting codes
FAILURE: $\widetilde{m} \neq m$

Classical error correction



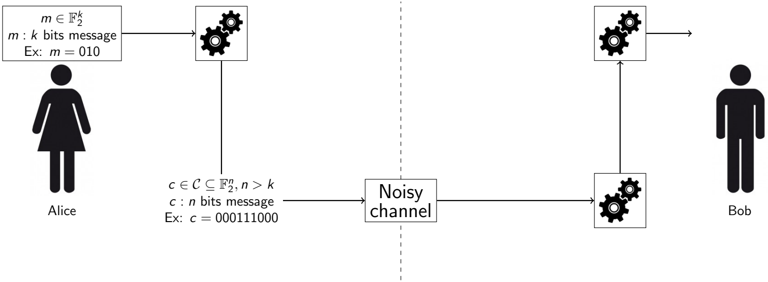
With error correcting codes

Classical error correction



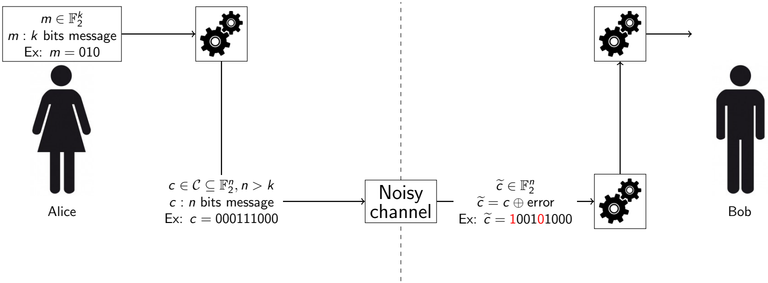
With error correcting codes

Classical error correction



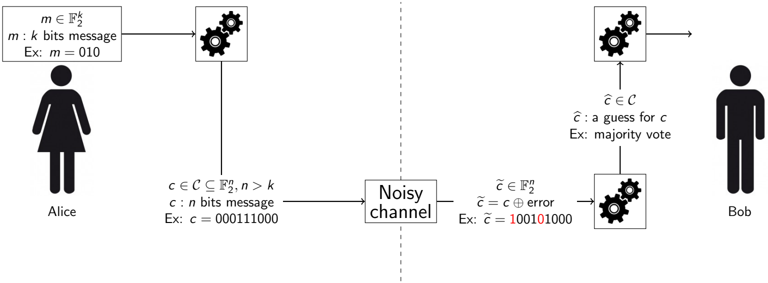
With error correcting codes

Classical error correction



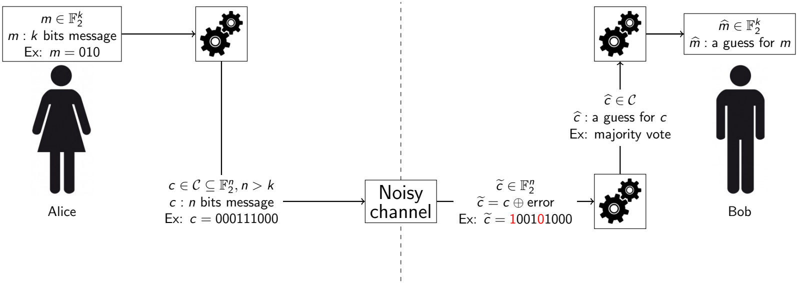
With error correcting codes

Classical error correction



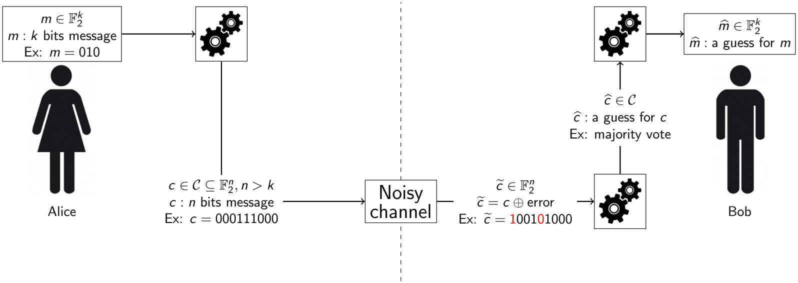
With error correcting codes

Classical error correction



With error correcting codes

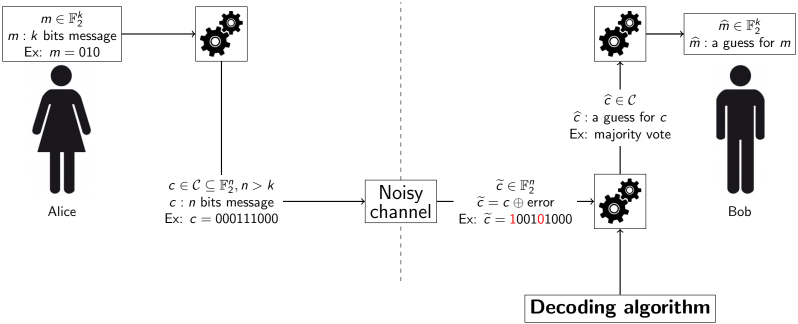
Classical error correction



With error correcting codes

Success condition: $\widehat{m} = m$ or equivalently $\widehat{c} = c$

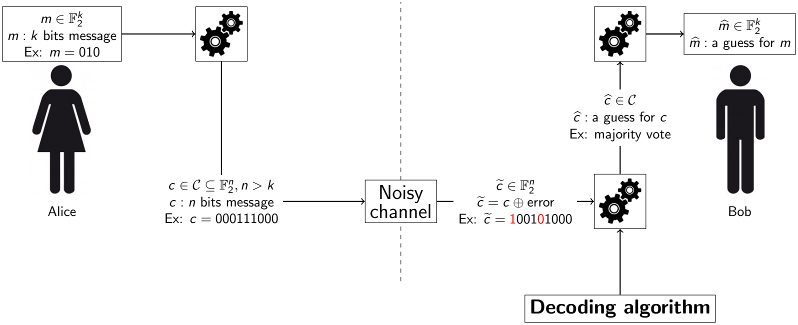
Classical error correction



With error correcting codes

Success condition: $\widehat{m} = m$ or equivalently $\widehat{c} = c$

Classical error correction



With error correcting codes

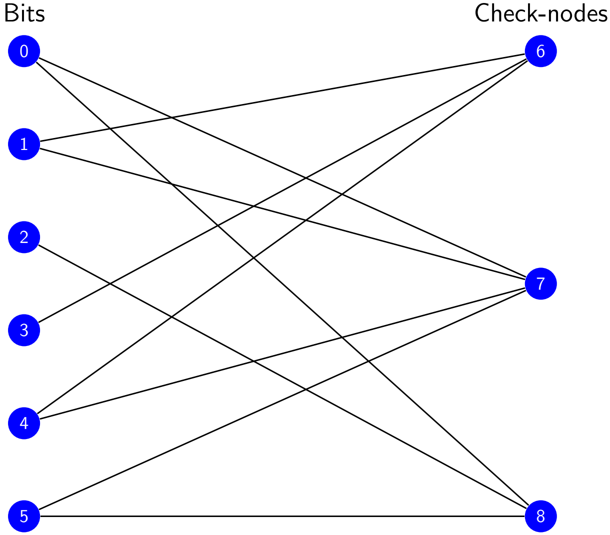
Success condition: $\widehat{m} = m$ or equivalently $\widehat{c} = c$

Definition: classical error correcting codes

- A $[n, k]$-error correcting code is a k-dimensional subspace of $\mathbb{F}_2^n$
- $H \in \mathcal{M}_{n-k,n}$ is a parity check matrix for a code $\mathcal{C}$ if $\mathcal{C} = \ker H$

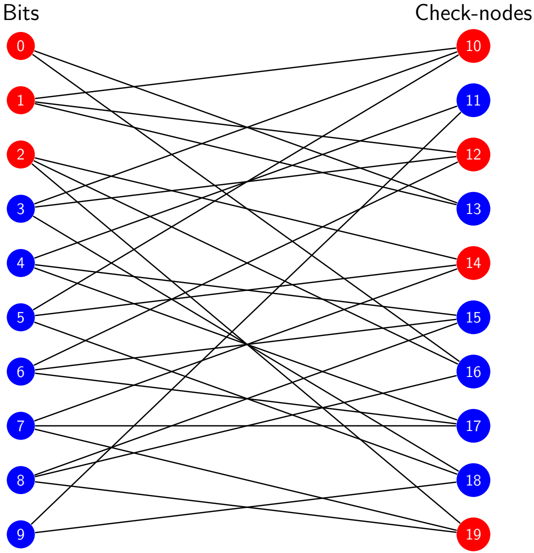
Factor graph of a code

$$\begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$



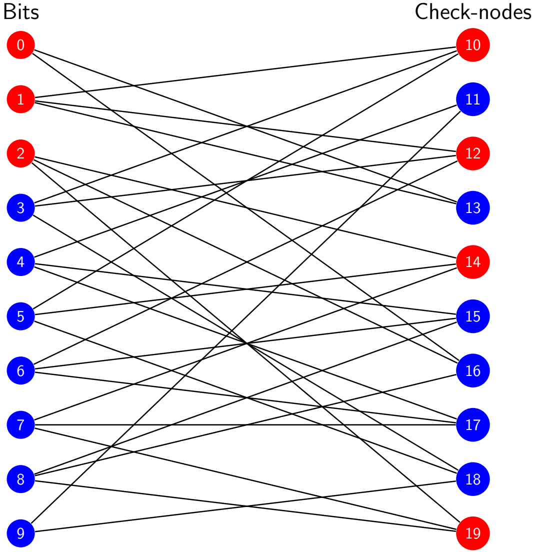
The bit-flip decoding algorithm

- Error:
  $e_0 = \{0, 1, 2\}$
- Unsatisfied check-nodes (syndrome):
  $\{10, 12, 14, 19\}$
- Satisfied check-nodes:
  $\{11, 13, 15, 16, 17, 18\}$

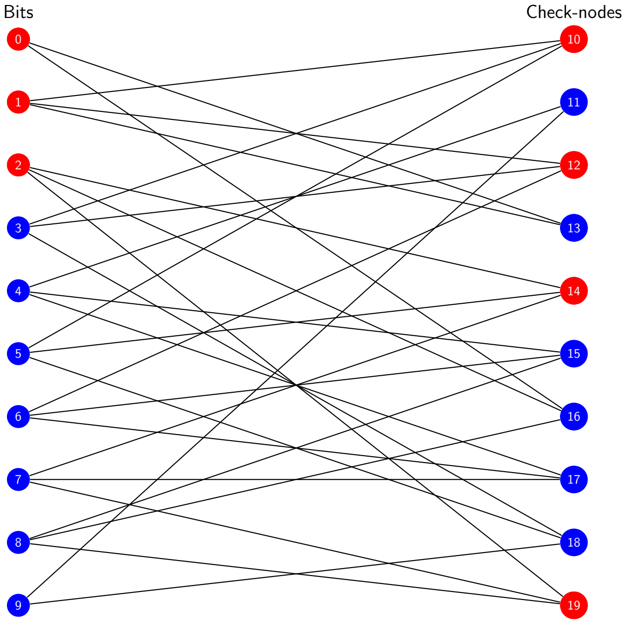


The bit-flip decoding algorithm

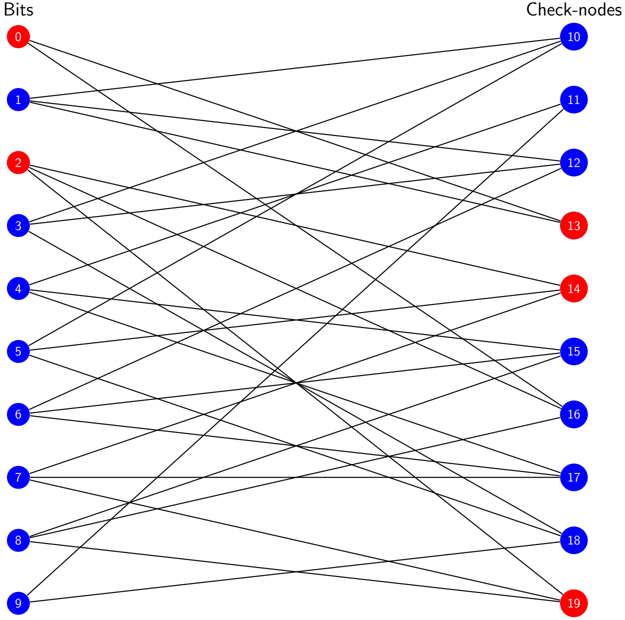
- Input: $\{10, 12, 14, 19\}$ (syndrome)
- The error e_0 is unknown
- Output: e
  a set of bits
- Success condition:
  $e = e_0$
- The algorithm flips a bit when it decreases the syndrome



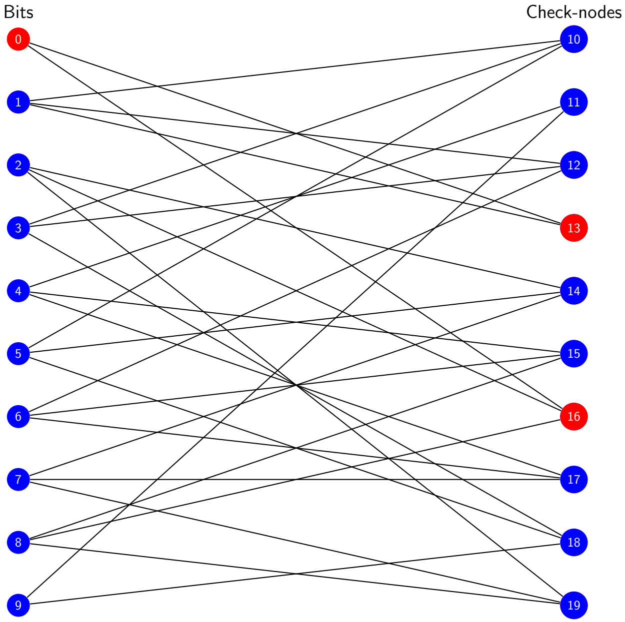
Decoding algorithm: first example



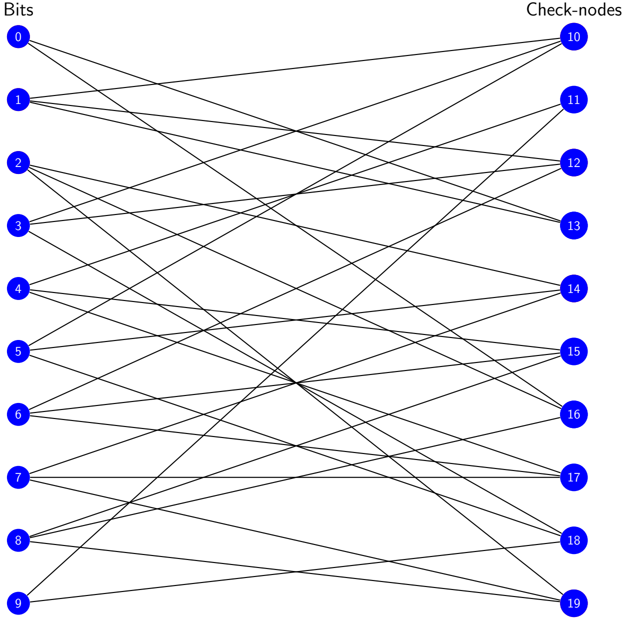
Decoding algorithm: first example



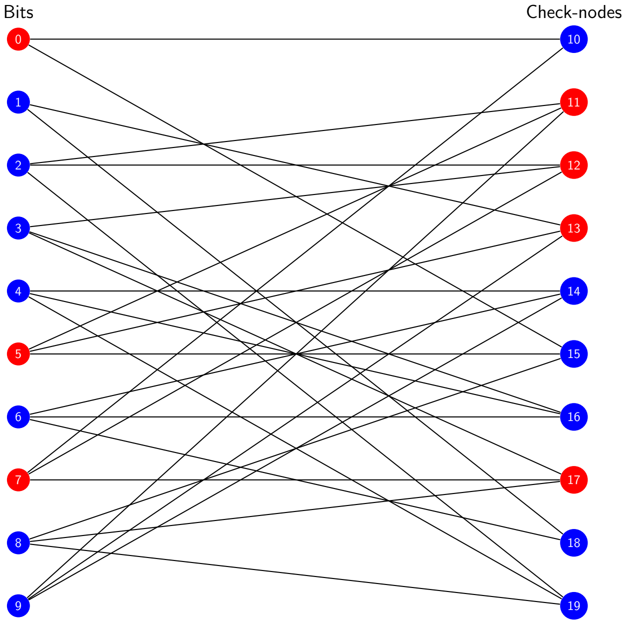
Decoding algorithm: first example



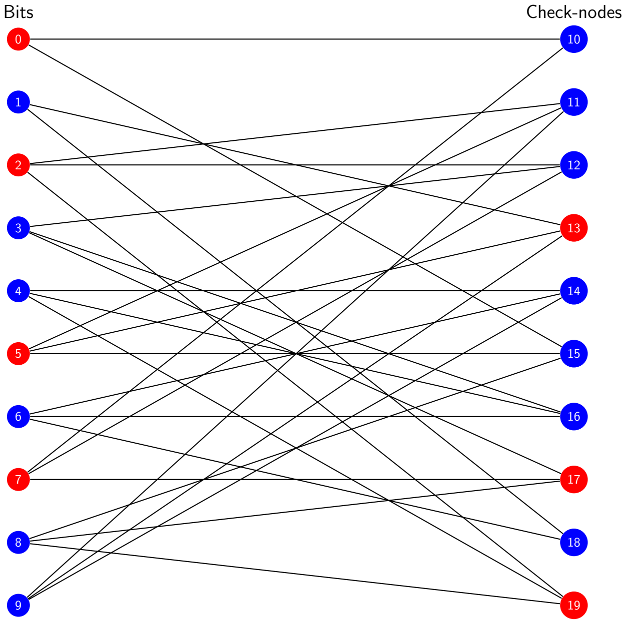
Decoding algorithm: first example



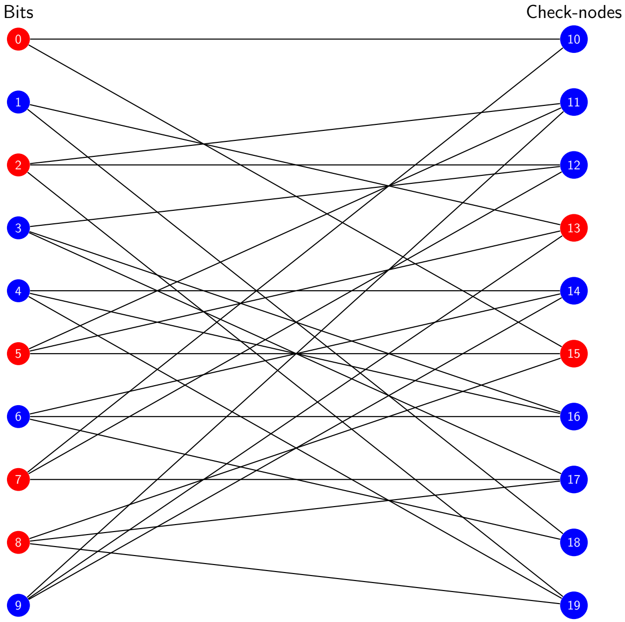
Decoding algorithm: second example



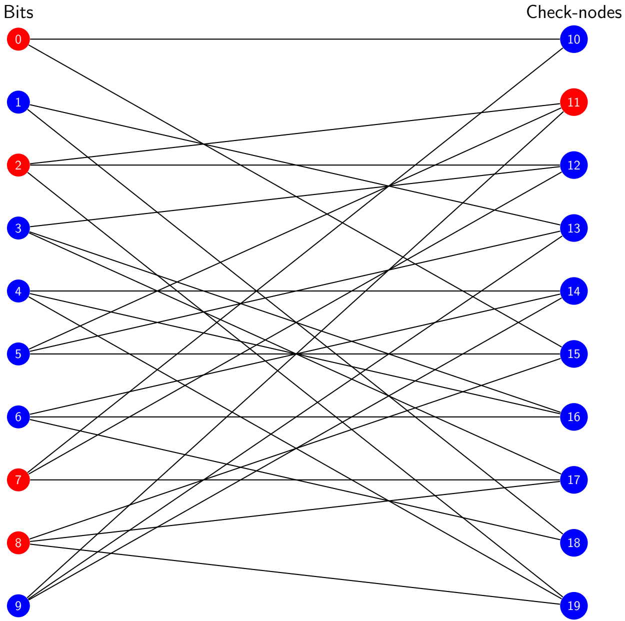
Decoding algorithm: second example



Decoding algorithm: second example

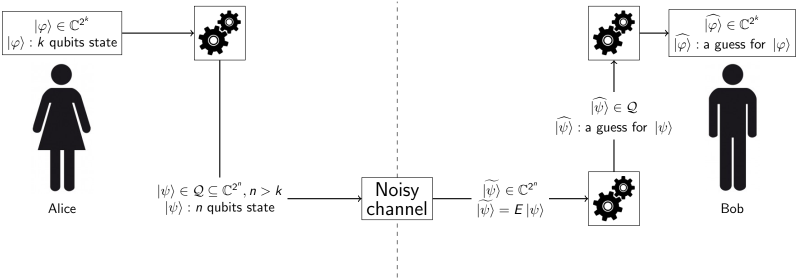


Decoding algorithm: second example



Outline

Quantum error correction



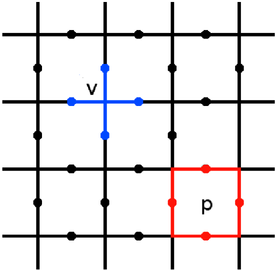
- Bit: $b \in \mathbb{F}_2$
- A $[n, k]$-code is a k-dimensional subspace of $\mathbb{F}_2^n$
- Classical error: Flip

- Qubit: $|b\rangle \in \mathbb{C}^2$, $\| \, |b\rangle \, \|_2 = 1$
- A $[\![n, k]\!]$-code is a 2^k-dimensional subspace of $\mathbb{C}^{2^n}$
- Quantum errors: X and Z

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

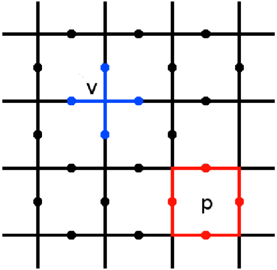
Example: the toric code

- n qubits on edges
- X-type generators associated with vertices
- Z-type generators associated with plaquettes
- $k = \#holes = 2$
- $d = \text{systole} = \sqrt{n/2}$
- Numerical simulations: 10% rate random errors are corrected



Example: the toric code

- n qubits on edges
- X-type generators associated with vertices
- Z-type generators associated with plaquettes
- $k = \#holes = 2$
- $d = \text{systole} = \sqrt{n/2}$
- Numerical simulations: 10% rate random errors are corrected



Adversarial errors VS Random errors:

- "Corrects adversarial errors of size up to $\Theta(\sqrt{n})$": any error of size up to $\Theta(\sqrt{n})$ is corrected
  $\rightarrow$ Link with the minimal distance
- "Corrects random errors of size $\Theta(n)$": an error where qubits are flipped with probability p independently is corrected with high probability
  $\rightarrow$ Framework of our result

Initial problem:

- The best known minimal distance for a constant rate LDPC code is $\Theta(\sqrt{n}\,\sqrt[4]{\log(n)})$ ([Freedman & Meyer & Luo '02])
- We want to correct random errors of size $\Theta(n)$ with very high probability

Initial problem:

- The best known minimal distance for a constant rate LDPC code is $\Theta(\sqrt{n}\ \sqrt[4]{\log(n)})$ ([Freedman & Meyer & Luo '02])
- We want to correct random errors of size $\Theta(n)$ with very high probability

Solution given by [Dennis & Kitaev & Landahl & Preskill '01], [Kovalev & Pryadko '13]:

- Use of graph percolation theory
- Given a constant rate LDPC code with minimal distance $d = \Omega(n^{\epsilon})$, the maximum likelihood decoder corrects random errors of size $\Theta(n)$ with very high probability

Initial problem:

- The best known minimal distance for a constant rate LDPC code is $\Theta(\sqrt{n}\ \sqrt[4]{\log(n)})$ ([Freedman & Meyer & Luo '02])
- We want to correct random errors of size $\Theta(n)$ with very high probability

Solution given by [Dennis & Kitaev & Landahl & Preskill '01], [Kovalev & Pryadko '13]:

- Use of graph percolation theory
- Given a constant rate LDPC code with minimal distance $d = \Omega(n^{\epsilon})$, the maximum likelihood decoder corrects random errors of size $\Theta(n)$ with very high probability

Remaining problem:

- The maximum likelihood decoder is exponential time in general

Efficient decoder

There is a polynomial time decoder which corrects random errors of size $\Theta(n)$ with very high probability

- Very high probability: $\mathbb{P}(\text{correction}) = 1 - o(1/n^c)$ for all $c \in \mathbb{N}$

Main Theorem

Quantum expander codes have an efficient decoder

Efficient decoder

There is a polynomial time decoder which corrects random errors of size $\Theta(n)$ with very high probability

- Very high probability: $\mathbb{P}(\text{correction}) = 1 - o(1/n^c)$ for all $c \in \mathbb{N}$

Main Theorem

Quantum expander codes have an efficient decoder

Consequence:

- We can apply [Gottesman, '13] with quantum expander codes
- Fault-tolerant quantum computation with constant overhead is possible

Outline

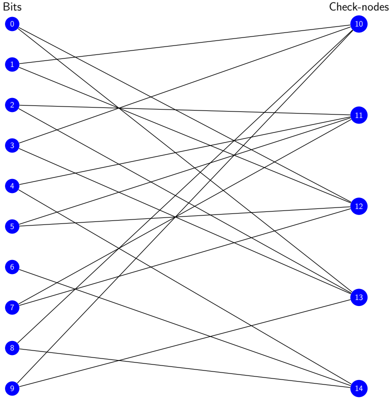
Summary of our contribution

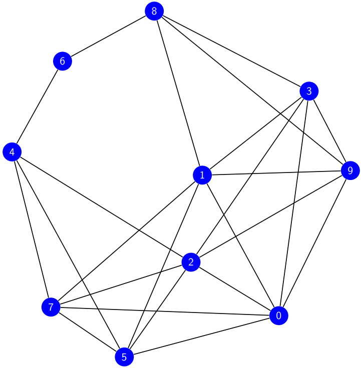
Question: What happens for random errors of size $\Theta(n)$?

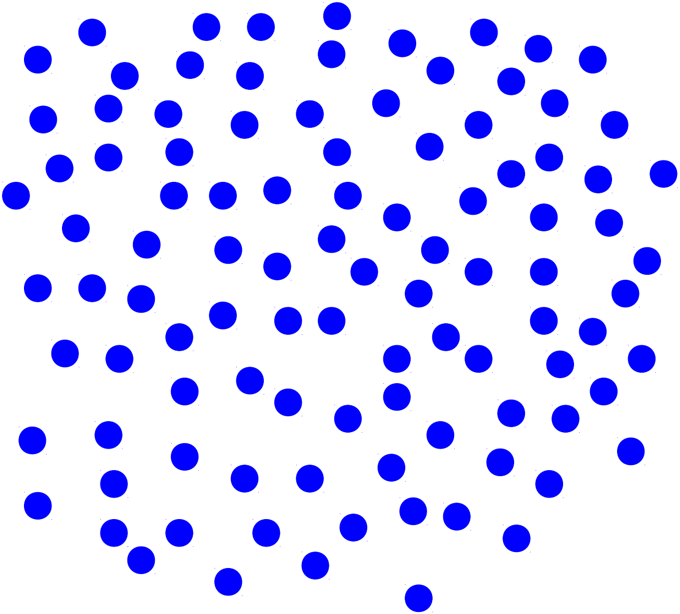
Theorem: what we proved

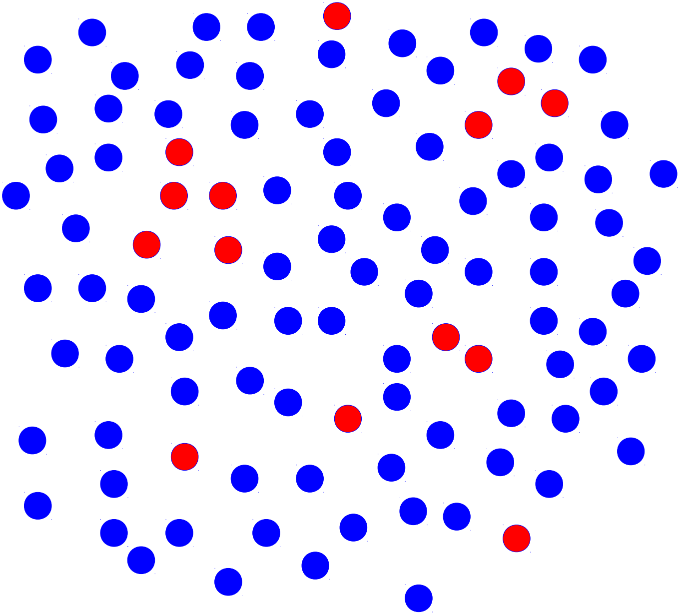
For a probability of error $p < p_{\text{th}}$:
$\mathbb{P}(\text{small-set-flip corrects the error}) = 1 - 1/e^{\Omega(\sqrt{n})}$

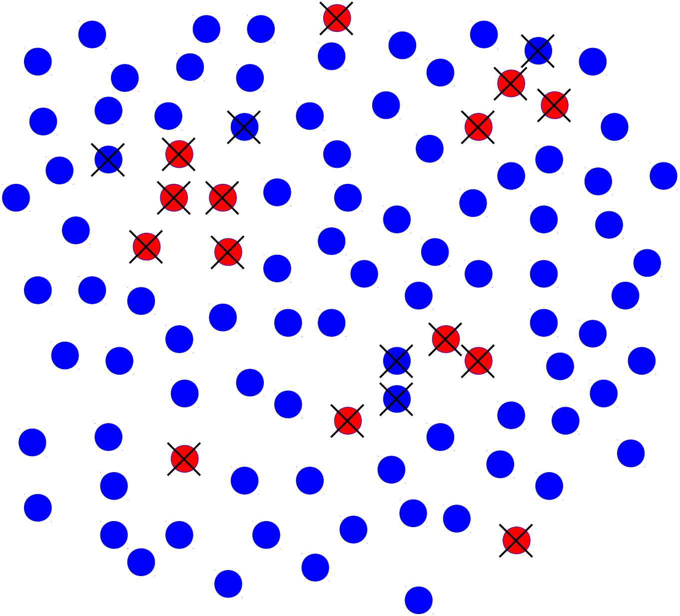
Idea. The algorithm is local with respect to the adjacency graph

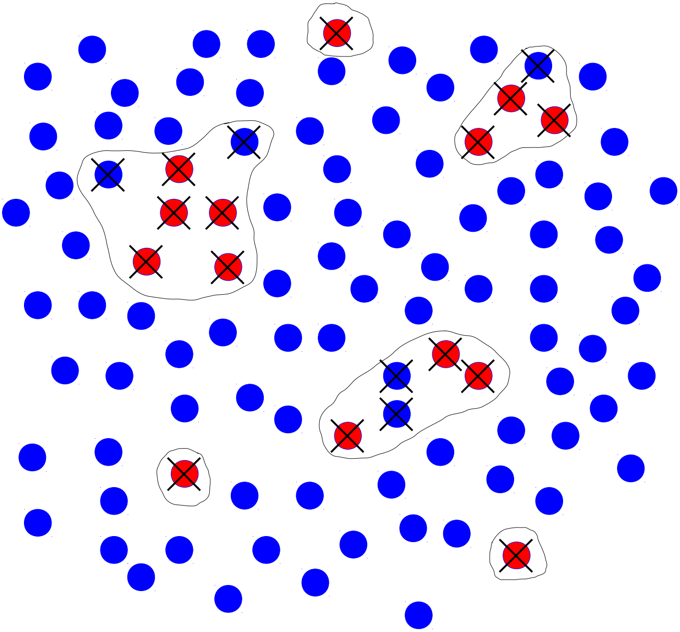


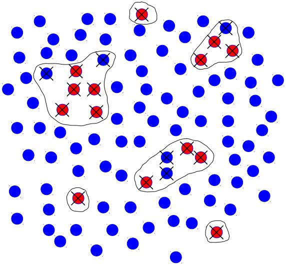












The number of flips is linear in the size of the initial error

Definition: α-subset, $\alpha \in (0, 1]$

X is an α-subset of E if $|X \cap E| \geq \alpha |X|$

- Each connected component X is an α-subset of $\{\text{red dots}\}$

Theorem: what we proved

For a probability of error $p < p_{\text{th}}$:
$\mathbb{P}(\text{small-set-flip corrects the error}) = 1 - 1/e^{\Omega(\sqrt{n})}$

Theorem: what we proved

For a probability of error $p < p_{\text{th}}$:
$\mathbb{P}(\text{small-set-flip corrects the error}) = 1 - 1/e^{\Omega(\sqrt{n})}$

Key lemma: percolation

Let $\alpha \in (0,1]$ and a probability of error $p < cst(\alpha, d)$.
With probability $1 - 1/e^{\Omega(\sqrt{n})}$:

- If X is a connected α-subset of the error then $|X| < c\sqrt{n}$

Theorem: what we proved

For a probability of error $p < p_{\text{th}}$:
$\mathbb{P}(\text{small-set-flip corrects the error}) = 1 - 1/e^{\Omega(\sqrt{n})}$

Key lemma: percolation

Let $\alpha \in (0, 1]$ and a probability of error $p < cst(\alpha, d)$.
With probability $1 - 1/e^{\Omega(\sqrt{n})}$:

- If X is a connected α-subset of the error then $|X| < c\sqrt{n}$

Sketch of the proof of the theorem:
Take a random error and run the small-set-flip algorithm. Let X be a connected component of the marked qubits:

- X is an α-subset of the error
- $|X| < c\sqrt{n}$
- X is corrected

This is true for any $X \rightarrow$ the entire error is corrected

Conclusion

Quantum expander codes:

- Are LDPC quantum codes
- Have a constant rate
- Have a good minimal distance: $d = \Theta(\sqrt{n})$

The decoder:

- Corrects any adversarial error of size up to $\Theta(\sqrt{n})$
- For a probability of error $p < p_{\text{th}}$: $\mathbb{P}(\text{correction}) = 1 - 1/e^{\Omega(\sqrt{n})}$

Corollary:

- Fault tolerant quantum computation with constant overhead is possible

Conclusion

Quantum expander codes:

- Are LDPC quantum codes
- Have a constant rate
- Have a good minimal distance: $d = \Theta(\sqrt{n})$

The decoder:

- Corrects any adversarial error of size up to $\Theta(\sqrt{n})$
- For a probability of error $p < p_{\text{th}}$: $\mathbb{P}(\text{correction}) = 1 - 1/e^{\Omega(\sqrt{n})}$

Corollary:

- Fault tolerant quantum computation with constant overhead is possible

Future work ($p_{\text{th}} \sim 10^{-16}$):

- Run simulations
- Improve our numerical value for the threshold

Conclusion

Quantum expander codes:

- Are LDPC quantum codes
- Have a constant rate
- Have a good minimal distance: $d = \Theta(\sqrt{n})$

The decoder:

- Corrects any adversarial error of size up to $\Theta(\sqrt{n})$
- For a probability of error $p < p_{\text{th}}$: $\mathbb{P}(\text{correction}) = 1 - 1/e^{\Omega(\sqrt{n})}$

Corollary:

- Fault tolerant quantum computation with constant overhead is possible

Future work ($p_{\text{th}} \sim 10^{-16}$):

- Run simulations
- Improve our numerical value for the threshold

Thank you for your attention

Known constructions of quantum LDPC codes

	k	Correction up to size	Efficient correction up to size
Toric code [Kit03]	2	$\Theta(\sqrt{n})$	$\Theta(\sqrt{n})$
Hyperbolic 2D [FML02]	$\Theta(n)$	$\Theta(\log n)$	$\Theta(\log n)$
Hyperbolic 4D [GL14], [Has13], [LL17]	$\Theta(n)$	$\Omega(n^{0.2}), \mathcal{O}(n^{0.3})$	$\Theta(\log n)$
Expander codes [TZ14], [LTZ15]	$\Theta(n)$	$\Theta(\sqrt{n})$	$\Theta(\sqrt{n})$

[Kit03] A Yu Kitaev. Fault-tolerant quantum computation by anyons, 2003

[FML02] Michael H Freedman, David A Meyer, and Feng Luo. Z2-systolic freedom and quantum codes, 2002

[GL14] Larry Guth and Alexander Lubotzky. Quantum error correcting codes and 4-dimensional arithmetic hyperbolic manifolds, 2014

[Has13] Matthew B Hastings. Decoding in hyperbolic spaces: Ldpc codes with linear rate and efficient error correction, 2013

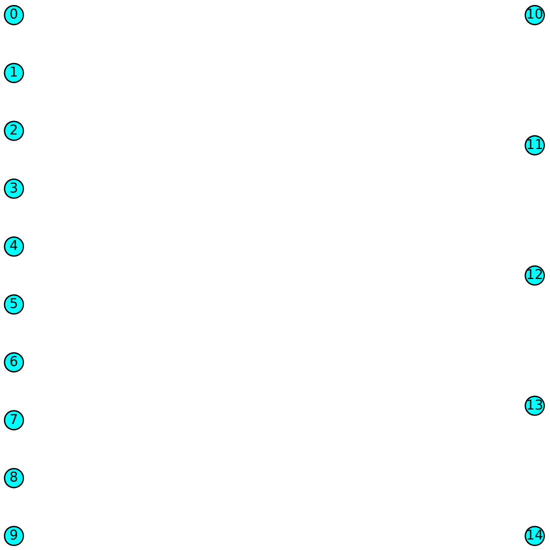
[LL17] Vivien Londe and Anthony Leverrier. Golden codes: quantum ldpc codes built from regular tessellations of hyperbolic 4-manifolds, 2017

[TZ14] Jean-Pierre Tillich and Gilles Zémor. Quantum ldpc codes with positive rate and minimum distance proportional to the square root of the blocklength, 2014

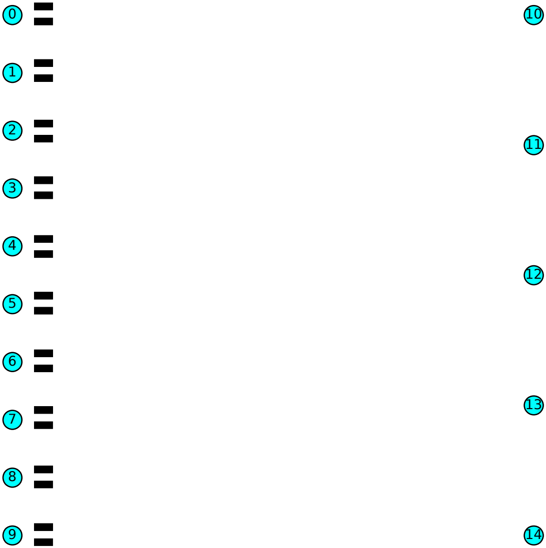
[LTZ15] Anthony Leverrier, Jean-Pierre Tillich, and Gilles Zémor. Quantum expander codes, 2015

$n = 10, m = 5, d_1 = 2, d_2 = \frac{n}{m} d_1 = 4$

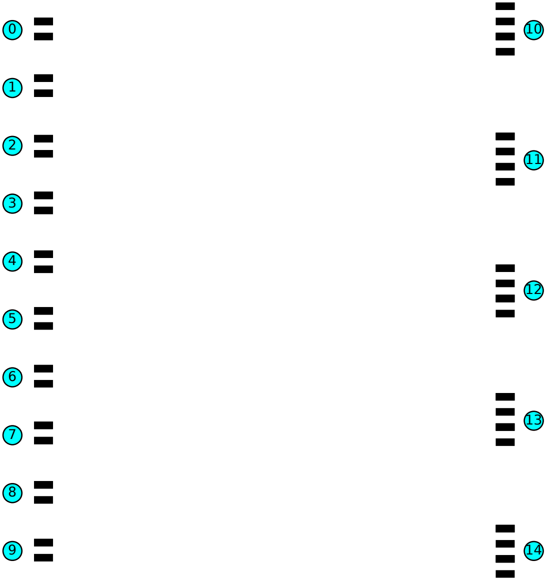
$n = 10, m = 5, d_1 = 2, d_2 = \frac{n}{m} d_1 = 4$



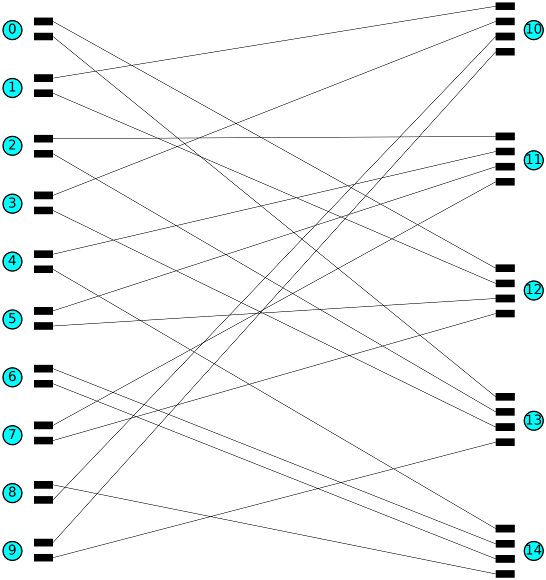
$n = 10, m = 5, d_1 = 2, d_2 = \frac{n}{m} d_1 = 4$



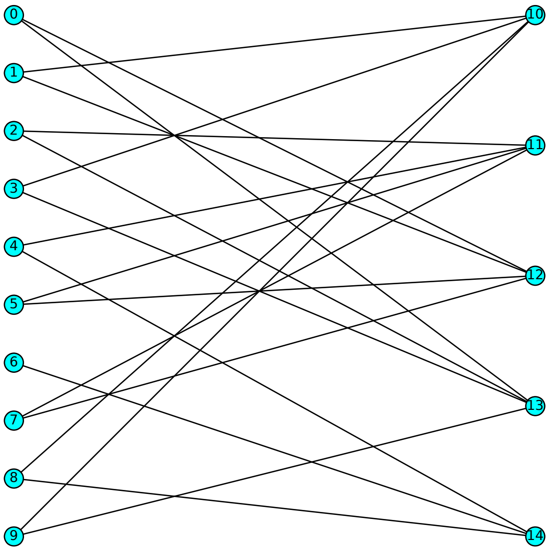
$n = 10, m = 5, d_1 = 2, d_2 = \frac{n}{m} d_1 = 4$



$n = 10, m = 5, d_1 = 2, d_2 = \frac{n}{m} d_1 = 4$



$n = 10, m = 5, d_1 = 2, d_2 = \frac{n}{m} d_1 = 4$



Stabilizer codes

Definition stabilizer codes: given a set $g_1, \ldots, g_{n-k}$ of commuting Pauli operators (product of X and Z Pauli matrices) on n qubits (called generators), we define a quantum code $\mathcal{Q}$ by:

$$\mathcal{Q} = \left\{ |\psi\rangle \in \mathbb{C}^{2^n} : g_1 |\psi\rangle = |\psi\rangle \cdots g_{n-k} |\psi\rangle = |\psi\rangle \right\}$$

Stabilizer codes

Definition stabilizer codes: given a set $g_1, \ldots, g_{n-k}$ of commuting Pauli operators (product of X and Z Pauli matrices) on n qubits (called generators), we define a quantum code $\mathcal{Q}$ by:

$$\mathcal{Q} = \left\{ |\psi\rangle \in \mathbb{C}^{2^n} : g_1 |\psi\rangle = |\psi\rangle \cdots g_{n-k} |\psi\rangle = |\psi\rangle \right\}$$

Parameters of a stabilizer code $[\![n, k, d]\!]$:

- $\mathcal{Q}$ encodes k logical qubits into n physical qubits
  i.e $\mathcal{Q}$ is a 2^k dimensional subspace of $\mathbb{C}^{2^n}$
- A logical error L is a Pauli operator such that $[L, g_i] = 0$ for all i and $L \notin \langle g_1, \ldots, g_{n-k} \rangle$
- The minimal distance d is the minimal weight of a logical error

The CSS construction

Definition [Calderbank & Shor '95], [Steane '95]

We can construct a quantum error correcting code using $\mathcal{C}_X$ and $\mathcal{C}_Z$ two classical error correcting codes such that $\mathcal{C}_X^{\perp} \subseteq \mathcal{C}_Z$

Each generator $g_1, \ldots, g_{n-k}$ of a CSS-code is either a product of Pauli X matrices or a product of Pauli Z matrices

The CSS construction

Definition [Calderbank & Shor '95], [Steane '95]

We can construct a quantum error correcting code using $\mathcal{C}_X$ and $\mathcal{C}_Z$ two classical error correcting codes such that $\mathcal{C}_X^\perp \subseteq \mathcal{C}_Z$

Each generator $g_1, \ldots, g_{n-k}$ of a CSS-code is either a product of Pauli X matrices or a product of Pauli Z matrices

Remark

The difficulty for constructing CSS code is to find two classical codes which are orthogonal

Hypergraph product codes [Tillich & Zémor '09]

The parity check matrix H of a classical code $\mathcal{C}$ satisfies $\mathcal{C} = \ker H$.
Let H be the parity check matrix of a classical code with constant rate and linear minimal distance.
We define the two classical codes $\mathcal{C}_X$ and $\mathcal{C}_Z$ by their parity check matrices:

$$H_X = (\mathbb{1} \otimes H, H^T \otimes \mathbb{1}) \qquad H_Z = (H \otimes \mathbb{1}, \mathbb{1} \otimes H^T)$$

Then $\mathcal{C}_X^{\perp} \subseteq \mathcal{C}_Z$

Hypergraph product codes [Tillich & Zémor '09]

The parity check matrix H of a classical code $\mathcal{C}$ satisfies $\mathcal{C} = \ker H$.
Let H be the parity check matrix of a classical code with constant rate and linear minimal distance.
We define the two classical codes $\mathcal{C}_X$ and $\mathcal{C}_Z$ by their parity check matrices:

$$H_X = (\mathbb{1} \otimes H, H^T \otimes \mathbb{1}) \qquad H_Z = (H \otimes \mathbb{1}, \mathbb{1} \otimes H^T)$$

Then $\mathcal{C}_X^\perp \subseteq \mathcal{C}_Z$

Definition

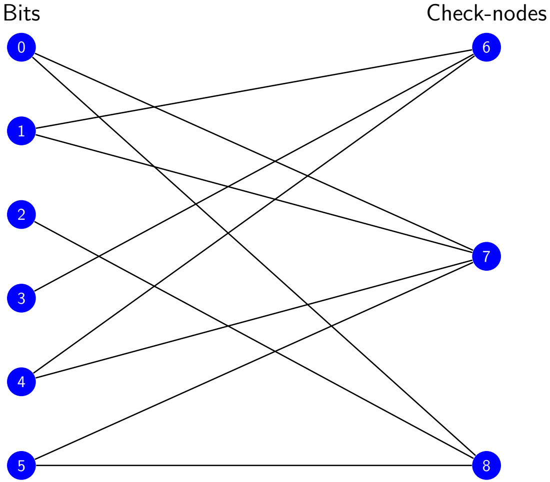
The hypergraph product is defined as $\mathrm{CSS}(\mathcal{C}_X, \mathcal{C}_Z)$.
It's a constant rate code with minimal distance $d = \Theta(\sqrt{n})$

- Freedom to choose H
- [Leverrier & Tillich & Zémor '15] chooses H as the parity check-matrix of a "classical expander code" ([Sipser & Spielman, '96])

Classical expander codes

The parity check matrix H of a classical code $\mathcal{C}$ satisfies $\mathcal{C} = \ker H$
H represented by a factor graph

$$\begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$



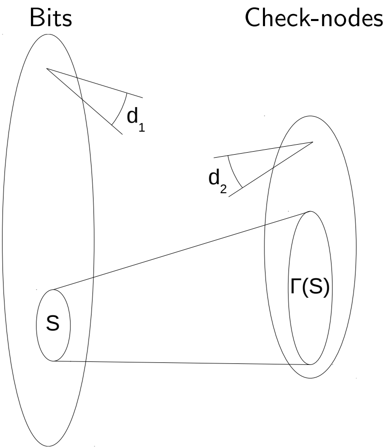
Classical expander codes

The parity check matrix H of a classical code $\mathcal{C}$ satisfies $\mathcal{C} = \ker H$
H represented by a factor graph

Definition of a (γ, δ) expander graph

For all $S \subseteq \{\text{Bits}\}$, if $|S| \leq \gamma n$ then:

$$|\Gamma(S)| \geq (1-\delta) d_1 |S|$$
$$|\Gamma(S)| \leq d_1 |S|$$



Expander graph
→ Parity check matrix
→ Classical expander code
→ Quantum expander code

Decoder for quantum expander codes

- **Classical case (bit-flip algorithm):**
  - As long as it is possible to flip a single bit to decrease the syndrome weight, flip this bit
  - This efficient algorithm corrects any adversarial error of size up to $\Theta(n)$ for classical expander codes [Sipser & Spielman, '96]

Decoder for quantum expander codes

- **Classical case (bit-flip algorithm):**
  - As long as it is possible to flip a single bit to decrease the syndrome weight, flip this bit
  - This efficient algorithm corrects any adversarial error of size up to $\Theta(n)$ for classical expander codes [Sipser & Spielman, '96]
- **Quantum case (small-set-flip algorithm):**
  - The "qubit-flip" algorithm doesn't work
  - Idea: try to flip several qubits at each step
  - As long as it is possible to flip a subset of a generator to decrease the syndrome weight, flip this subset

Decoder for quantum expander codes

- **Classical case (bit-flip algorithm):**
  - As long as it is possible to flip a single bit to decrease the syndrome weight, flip this bit
  - This efficient algorithm corrects any adversarial error of size up to $\Theta(n)$ for classical expander codes [Sipser & Spielman, '96]
- **Quantum case (small-set-flip algorithm):**
  - The "qubit-flip" algorithm doesn't work
  - Idea: try to flip several qubits at each step
  - As long as it is possible to flip a subset of a generator to decrease the syndrome weight, flip this subset

Theorem [Leverrier & Tillich & Zémor '15]

This efficient algorithm corrects any adversarial error of size up to $\Theta(\sqrt{n})$ for quantum expander codes