



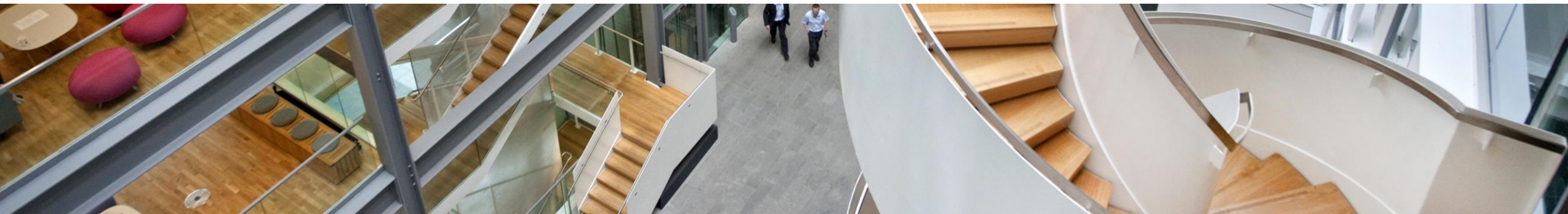
Quantum Error Correction: Surface Codes

IQC 2024 Lecture 29

Instructor: Joschka Roffe, joschka.roffe@ed.ac.uk



THE UNIVERSITY of EDINBURGH
informatics



Recap: Stabiliser Codes

The Pauli Group: $\mathcal{P}^{\otimes n}$ is the Pauli-group $(\pm I, \pm iI, \pm X, \pm Y, \pm Z, \pm iX, \pm iY, \pm iZ)$ over n qubits. E.g., $X_1 I_2 Y_3 \in \mathcal{P}^{\otimes 3}$.

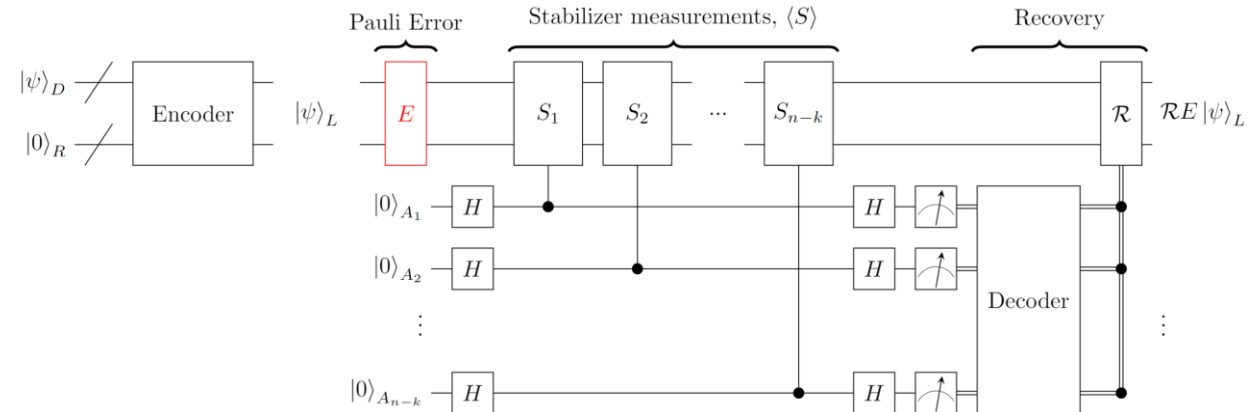
The stabilisers $\mathcal{S} \subset \mathcal{P}^{\otimes N}$ of a quantum error correction code simultaneously act as the identity on the logical state $|\psi\rangle_L$ such that:

$$G_i |\psi\rangle_L = (+1) |\psi\rangle_L \quad \text{for all } G_i \in \mathcal{S}$$

To detect errors, a generating set $\langle S \rangle \subset \mathcal{S}$ is measured using the Hadamard test syndrome extraction gadgets (as described in Lecture 25).

Minimal generating set of stabilisers

A generating $\langle S \rangle$ is minimal if all operators in S are independent. E.g., $[Z_1 Z_2, Z_2 Z_3, Z_1 Z_3]$ is not a generating set, as $Z_1 Z_3 = (Z_1 Z_2)(Z_2 Z_3)$.



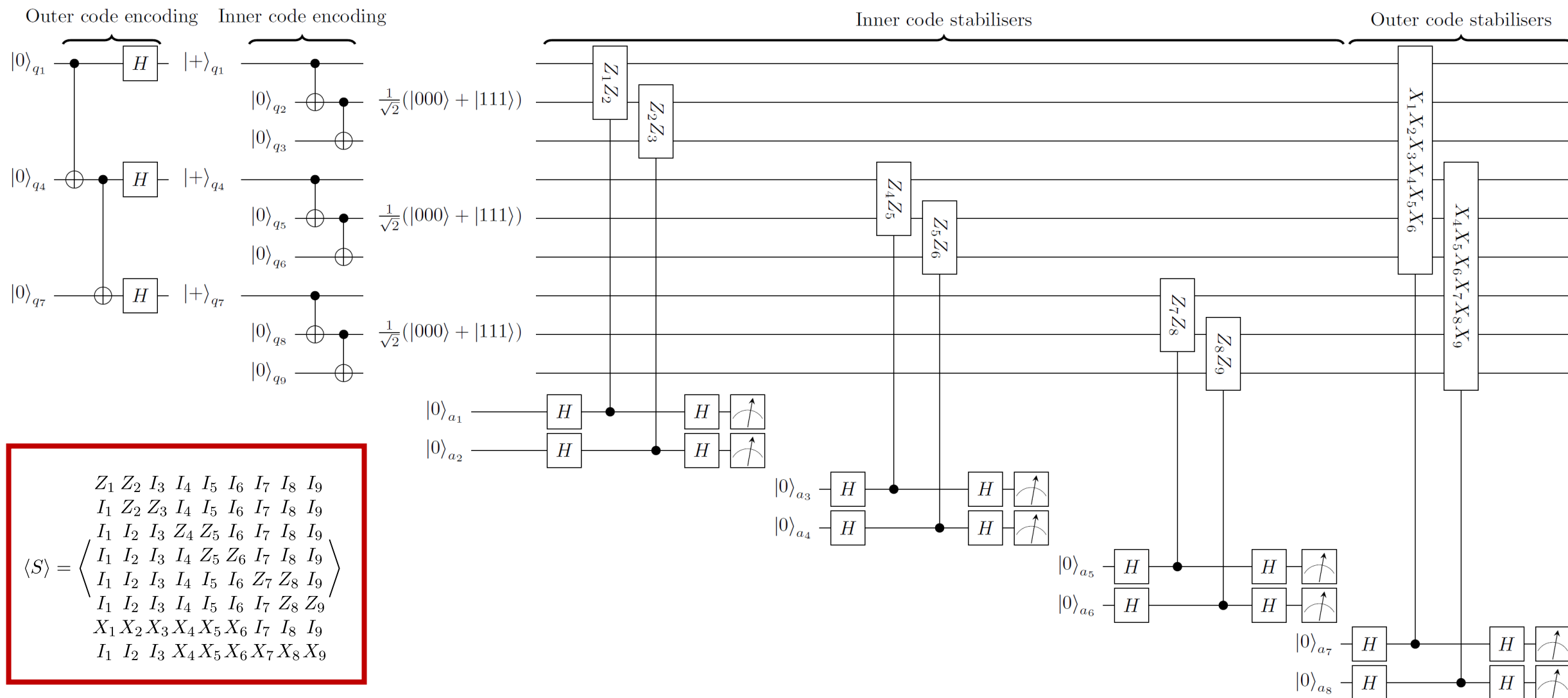
Stabiliser code syndromes

Measurement of generator S_i will yield outcome:

- $s_i = 0$: if the error commutes with the generator, $[E, S_i] = 0$
- $s_i = 1$: if the error anti-commutes with the generator, $\{E, S_i\} = 0$

The **syndrome** is the binary string obtained by concatenating all of the generator measurement outcomes: $s = s_1 s_2 \dots s_m$, where m is the size of the generating set, $m = |S|$.

Recap: The [9,1,3] Shor Code



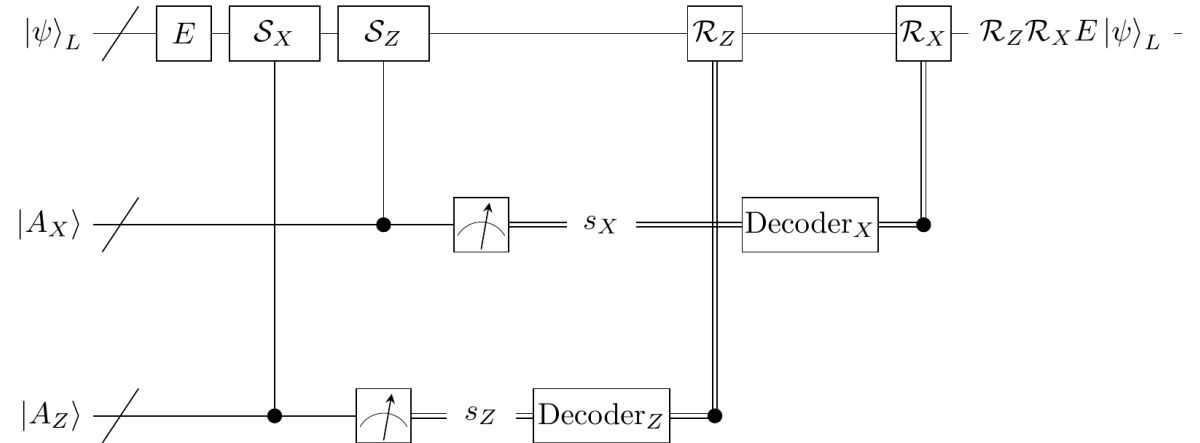
Calderbank-Shor-Steane (CSS) Codes

Note that the stabilisers for the nine qubit Shor code can be partitioned into two types:

- X-type stabilisers S_X that detect phase-flips
- Z-type stabilisers S_Z that detect bit-flips.

$$\langle S \rangle = \left\langle \begin{array}{cccccccccc} Z_1 & Z_2 & I_3 & I_4 & I_5 & I_6 & I_7 & I_8 & I_9 \\ I_1 & Z_2 & Z_3 & I_4 & I_5 & I_6 & I_7 & I_8 & I_9 \\ I_1 & I_2 & I_3 & Z_4 & Z_5 & I_6 & I_7 & I_8 & I_9 \\ I_1 & I_2 & I_3 & I_4 & Z_5 & Z_6 & I_7 & I_8 & I_9 \\ I_1 & I_2 & I_3 & I_4 & I_5 & I_6 & Z_7 & Z_8 & I_9 \\ I_1 & I_2 & I_3 & I_4 & I_5 & I_6 & I_7 & Z_8 & Z_9 \\ X_1 & X_2 & X_3 & X_4 & X_5 & X_6 & I_7 & I_8 & I_9 \\ I_1 & I_2 & I_3 & X_4 & X_5 & X_6 & X_7 & X_8 & X_9 \end{array} \right\rangle$$

Quantum error correction codes that partition stabilisers in this way are referred to as **Calderbank-Shor-Steane (CSS)** codes. The $[[4,2,2]]$ code, the $[[7,1,3]]$ Steane code are also examples of CSS-type stabiliser codes.



We think of a CSS code as a combination of two separate error correction protocols: one for bit-flips and one for phase-flips. The full stabiliser group is generated by:

$$S = S_X \cup S_Z = \langle S_X \cup S_Z \rangle$$

Quantum Error Correction Code Constructions

The Challenge of QEC Code design

Quantum code design for CSS codes is complicated by the fact that the stabiliser group must be Abelian. I.e.,

$$[G_i^X, G_j^Z] = 0 \text{ for all } G_i^X \in \mathcal{S}_X \text{ and } G_j^Z \in \mathcal{S}_Z$$

Finding generating sets of stabilisers \mathcal{S}_X and \mathcal{S}_Z is not straightforward. E.g., we can't just translate classical error correction codes into the \mathcal{S}_X and \mathcal{S}_Z of a quantum error correction codes.

Up to this point, all the of the code constructions we have looked at – the $[[4,2,2]]$ code, the Steane code, and the Shor Code – have been *hand crafted*.

The goal is now to develop **systematics procedures** for designing large-scale CSS stabiliser codes.

Requirements for a Quantum Code Construction

A systematic approach for CSS code design must address the following:

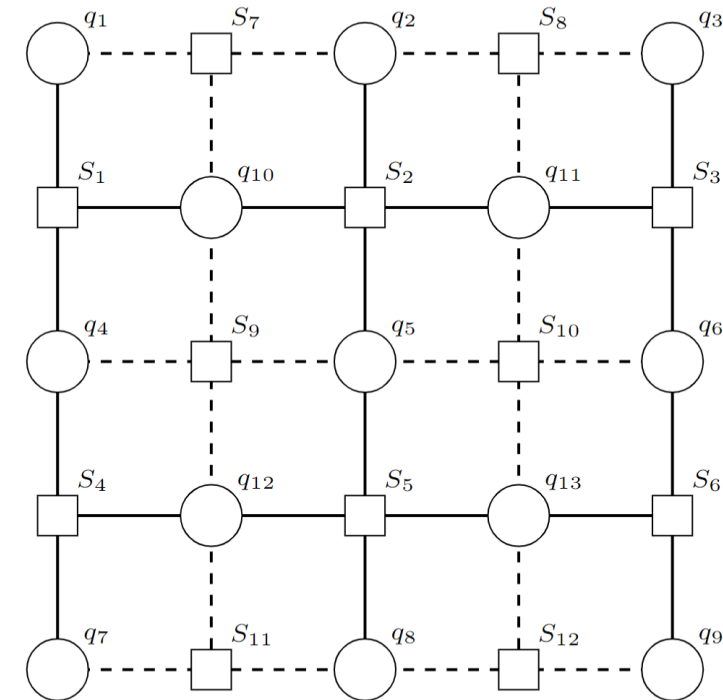
- **Generating Stabiliser Sets:** Provide a method to construct \mathcal{S}_X and \mathcal{S}_Z such that they are guaranteed to commute.
- **Scalability:** Ensure that the code distance d increases with the size of the system.



The Surface Code

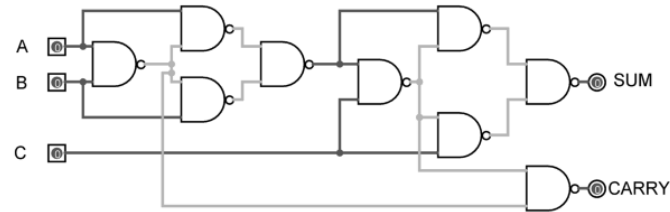
The **surface** code is the leading construction for experimental quantum error correction. It has two major advantages:

- It can be implemented using nearest-neighbour CNOT gates between qubits arranged in a two-dimensional grid. This is particularly useful for many qubit hardware types – e.g. superconducting qubits – where long-range gates are difficult to implement with high-fidelity.
- The distance can be straightforwardly scaled by increasing the size of the qubit grid.



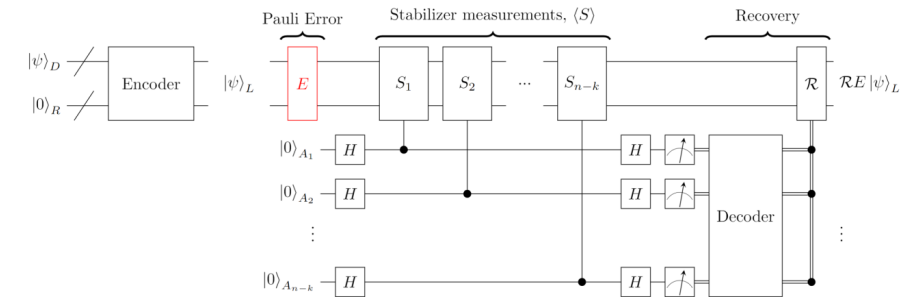
The Tanner (connectivity) graph for the $d = 3$ surface code.

Classical vs. Quantum Circuits



Classical circuit

- Provides information about what operations are performed.
- Tells us exactly how to wire together the circuit.

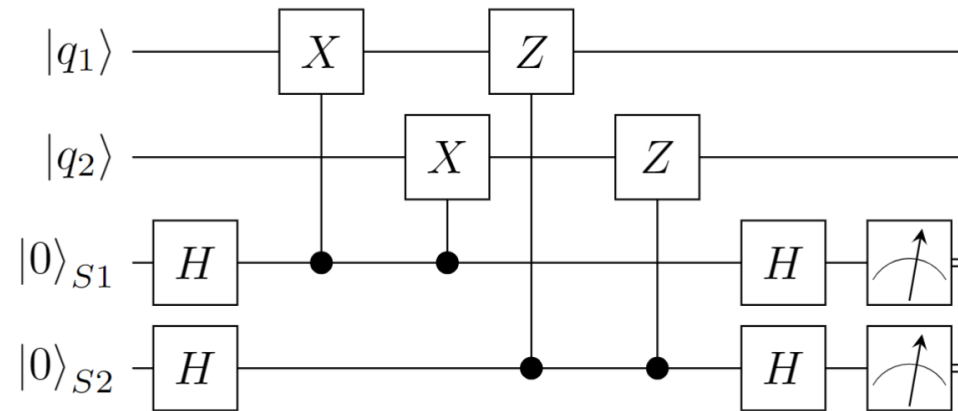
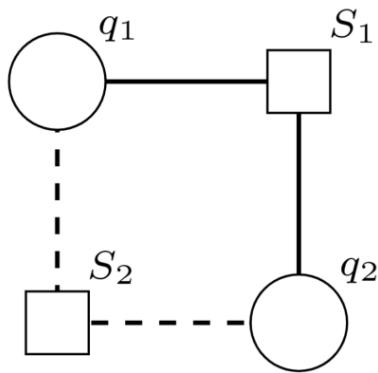


Quantum Circuit

- Provides information about what operations are performed.
- Wires represent the passage of time. I.e., they do not actually correspond to wires and tell us nothing about the qubit connectivity.

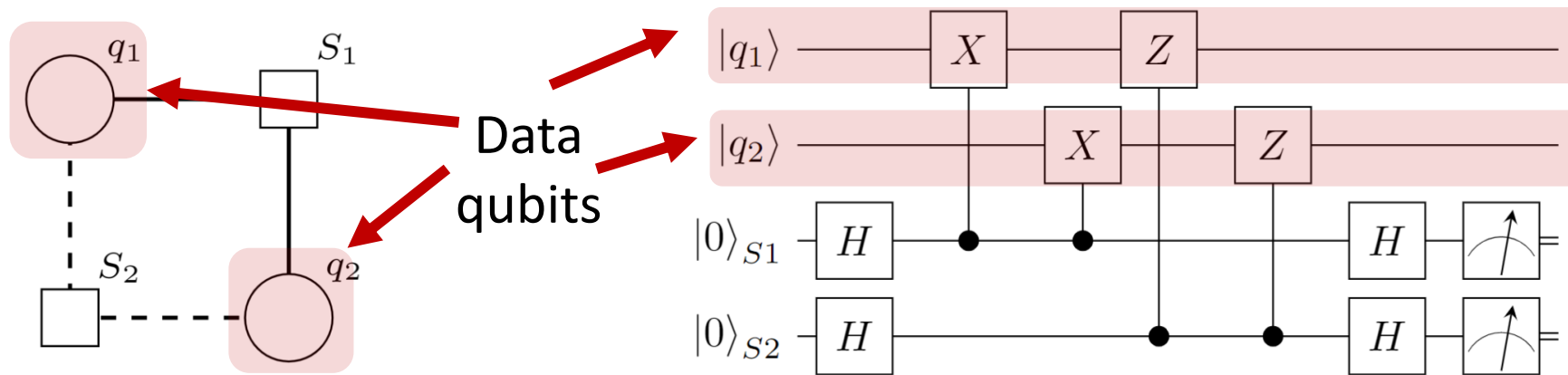
The Tanner Graph Representation of QEC

Tanner graphs provide a graphical representation of quantum error correction that allows us to design codes in a way that is *connectivity aware*.



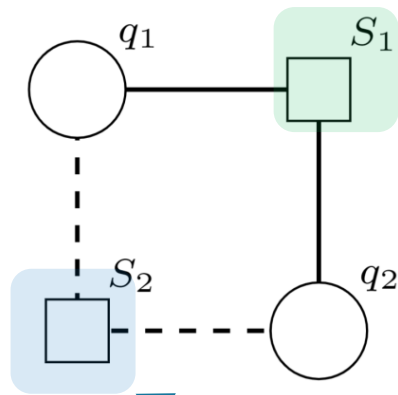
The Tanner Graph Representation of QEC

Tanner graphs provide a graphical representation of quantum error correction that allows us to design codes in a way that is *connectivity aware*.

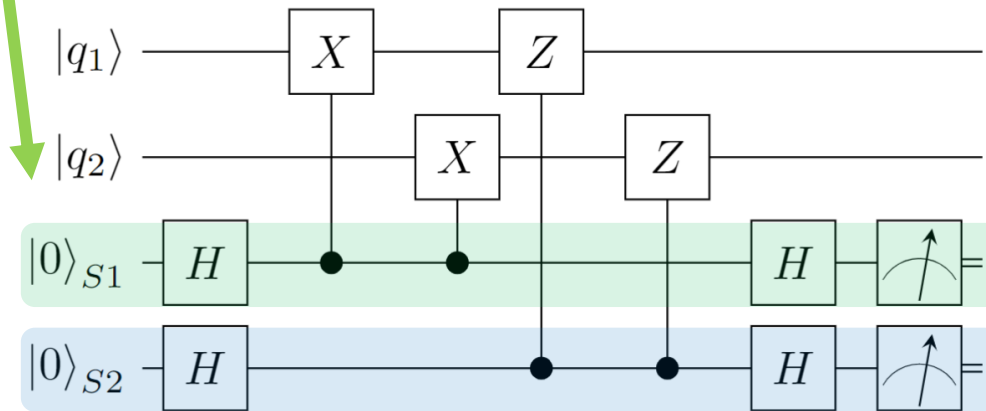


The Tanner Graph Representation of QEC

Tanner graphs provide a graphical representation of quantum error correction that allows us to design codes in a way that is *connectivity aware*.

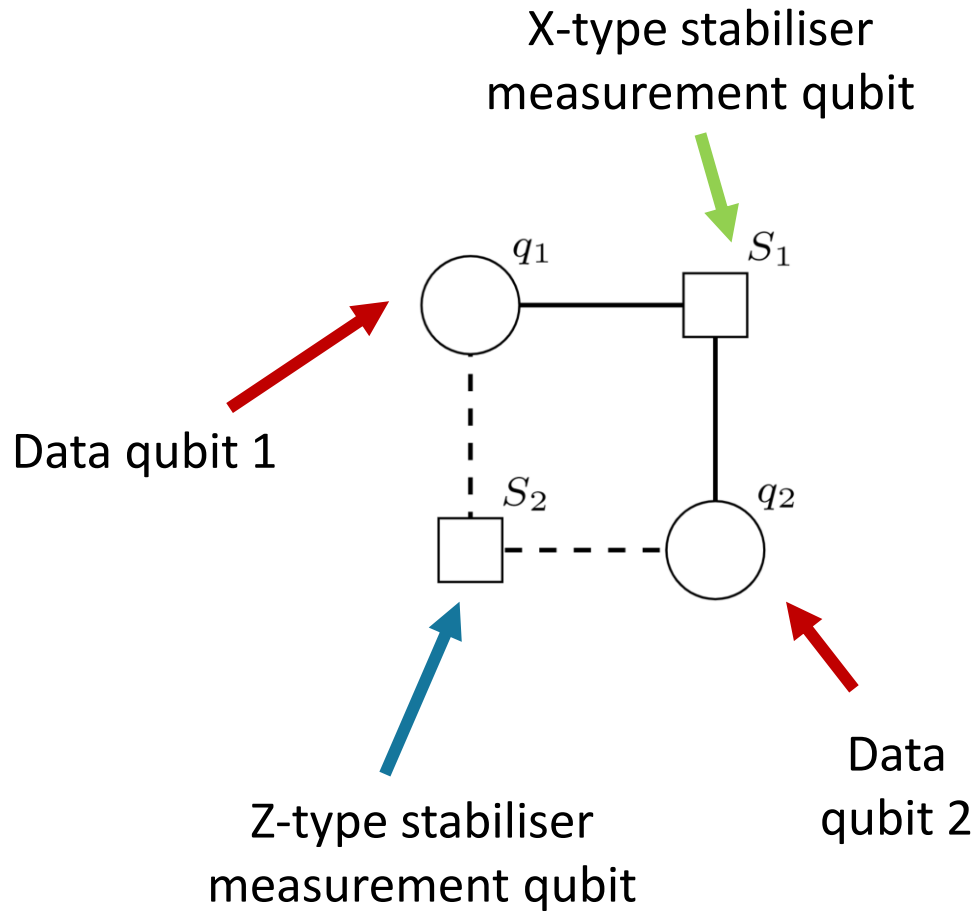


X-type stabiliser
measurement qubit



Z-type stabiliser
measurement qubit

The Surface Code 4-Cycle



Stabilisers:

$$\langle S \rangle = \langle S_X \cup S_Z \rangle = \langle S_1 \cup S_2 \rangle = \langle Z_1 Z_2, X_1 X_2 \rangle$$

Both stabilisers intersect on two-qubits: they commute!

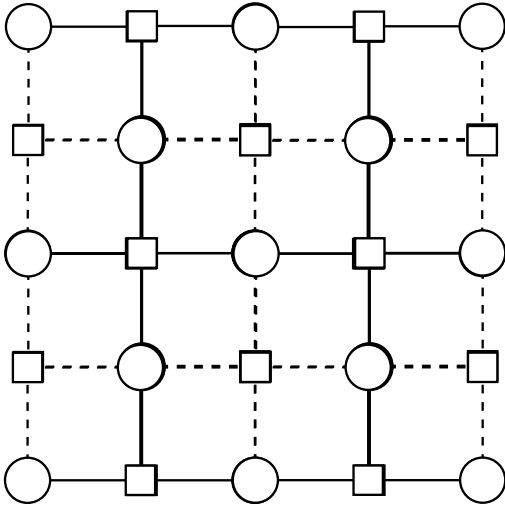
Logical qubit count

$$k = n - |S| = 2 - 2 = 0$$

The surface code four-cycle does not encode information!



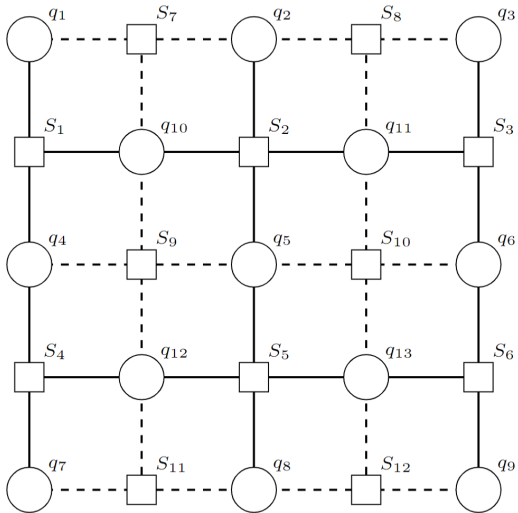
The Surface Code



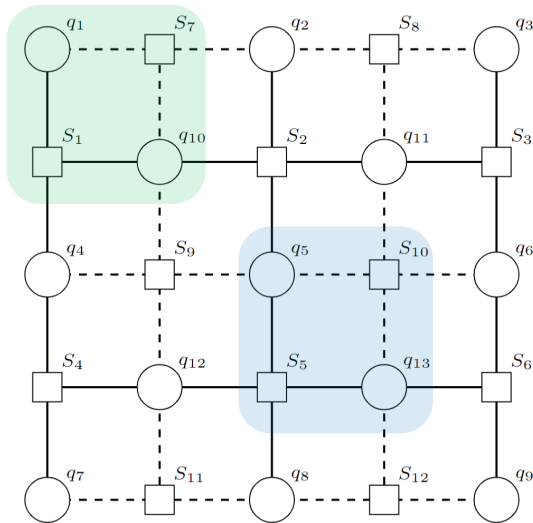
Larger surface codes can be constructed by tiling four-cycles together.

The Surface Code

Larger surface codes can be constructed by tiling four-cycles together.



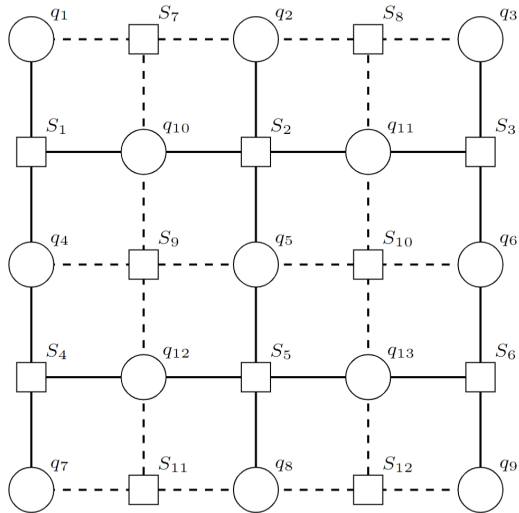
The $d = 3$ Surface Code



Larger surface codes can be constructed by tiling four-cycles together.

The construction guarantees that any two X- and Z-type stabiliser always intersect non-trivially on an even number of qubits. \Rightarrow The stabilisers always commute!

Surface codes: logical qubit count



This surface code has $n = 13$ physical (data) qubits

There are 12 auxiliary qubits measuring 12 independent stabiliser generators. \Rightarrow
 $rank(\mathcal{S}) = |\mathcal{S}| = 12$

Logical qubit count:

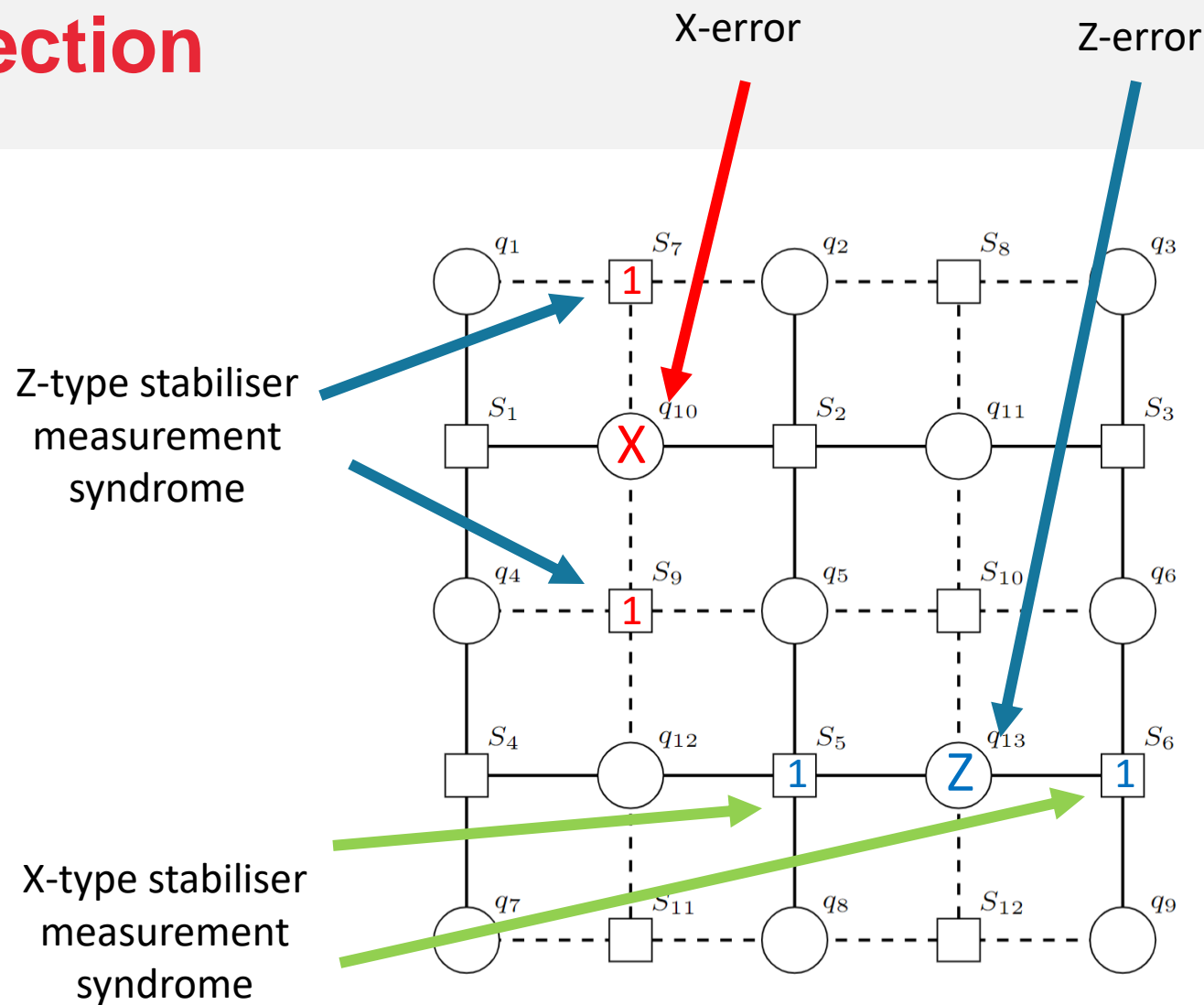
$$k = n - |\mathcal{S}| = 1$$

For a surface patch of any size, the number of stabiliser generators is $|\mathcal{S}| = n - 1$.

This means all surface codes (other than the surface code four-cycle) encode $k = 1$ logical qubits

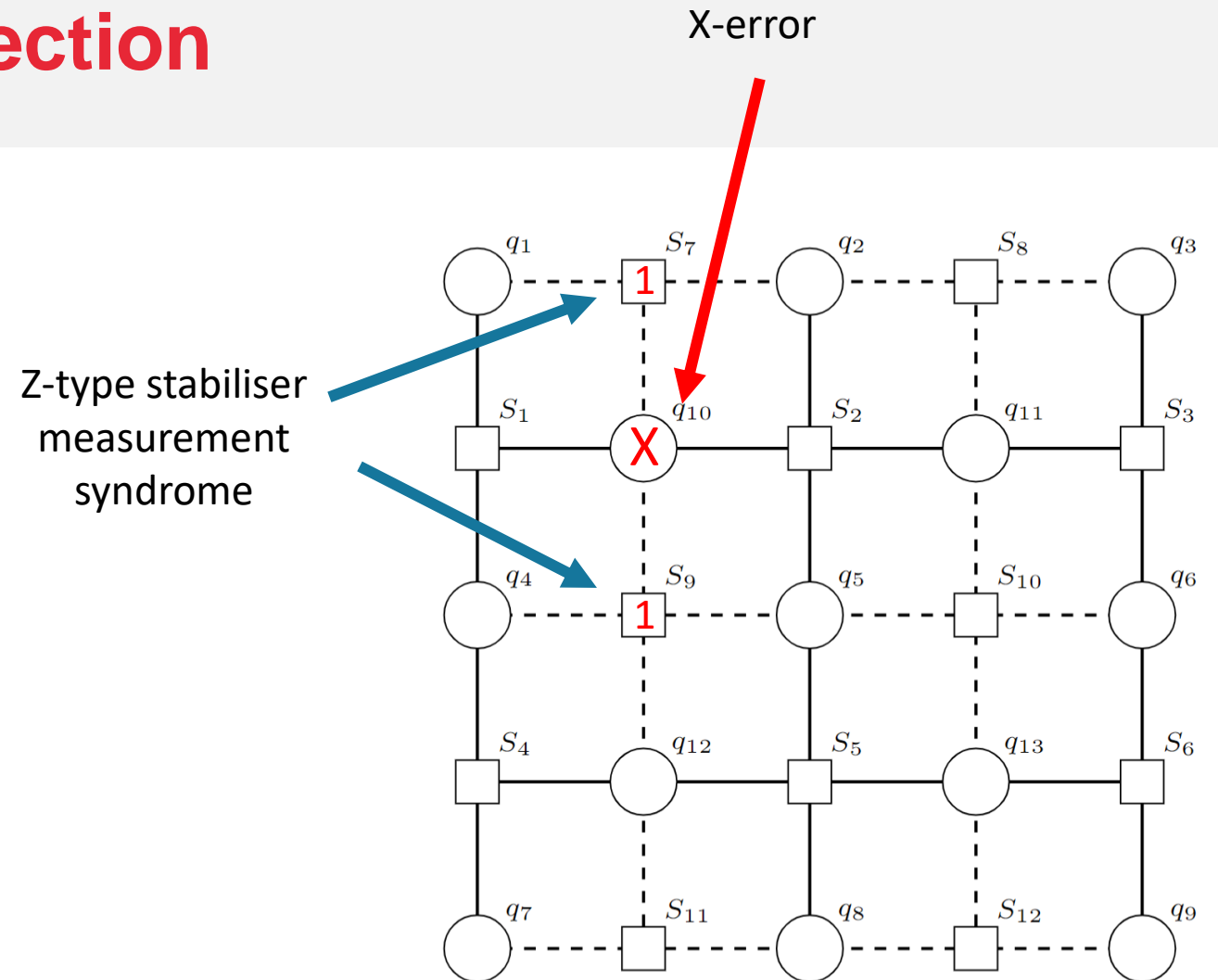
Surface codes: error detection

Surface codes are CSS codes:
X- and Z-type errors are
detected by different
stabilisers.



Surface codes: error detection

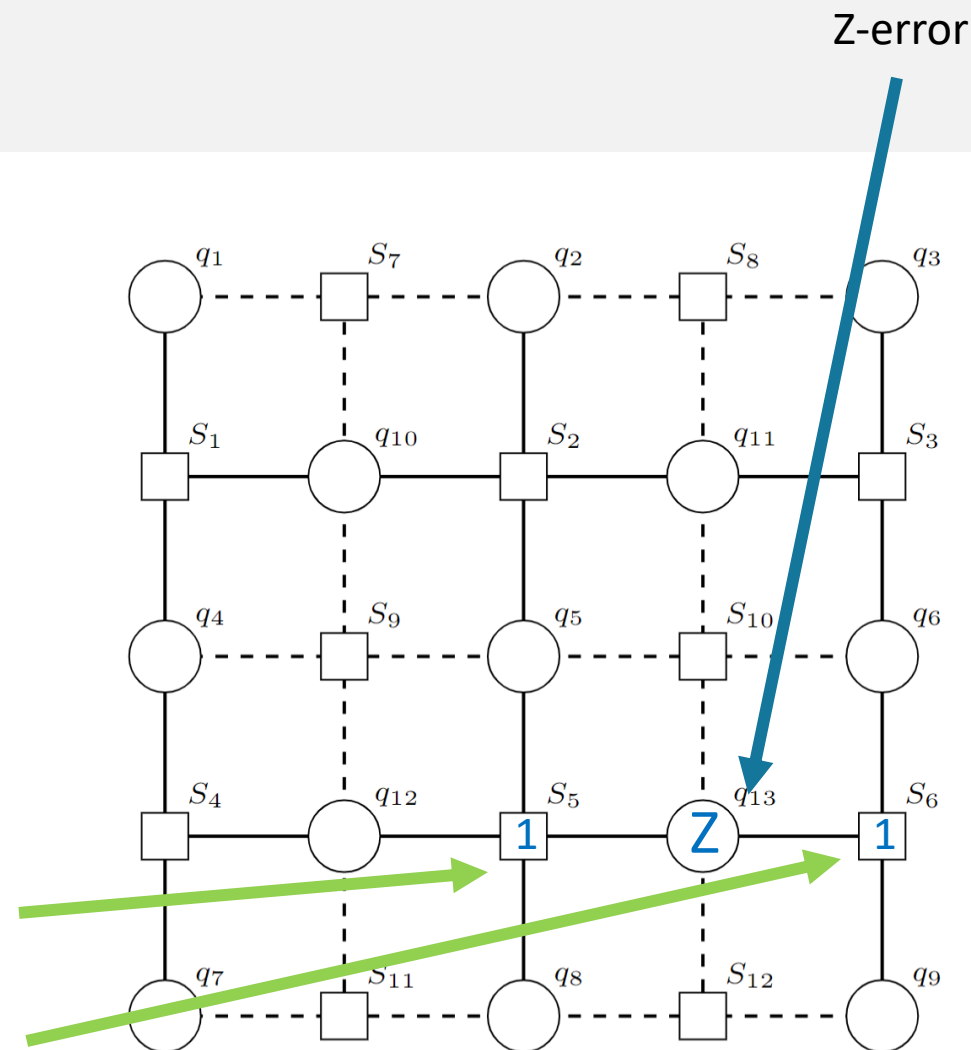
Surface codes are CSS codes:
X- and Z-type errors are
detected by different
stabilisers.



Surface codes: error detection

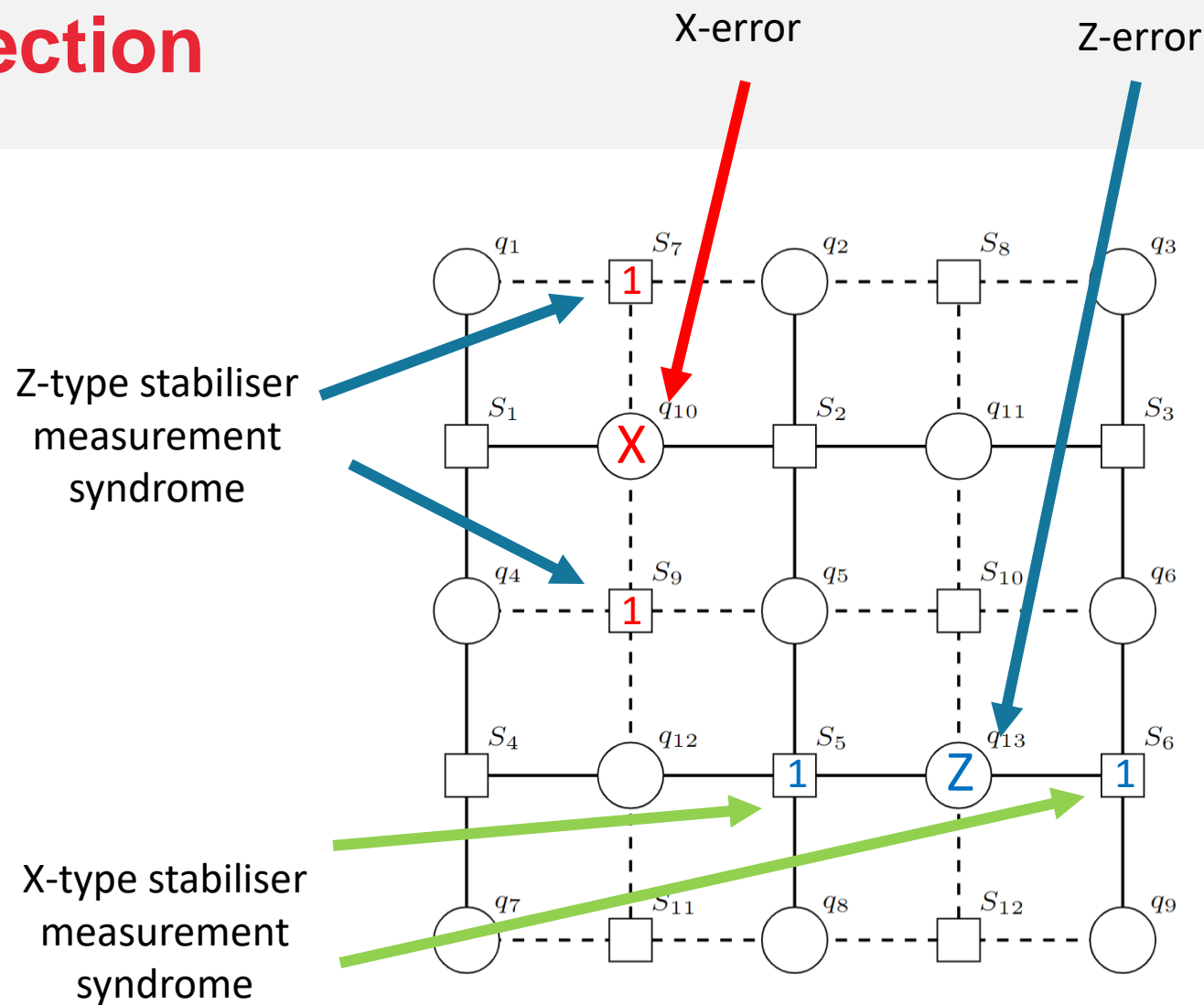
Surface codes are CSS codes:
X- and Z-type errors are
detected by different
stabilisers.

X-type stabiliser
measurement
syndrome

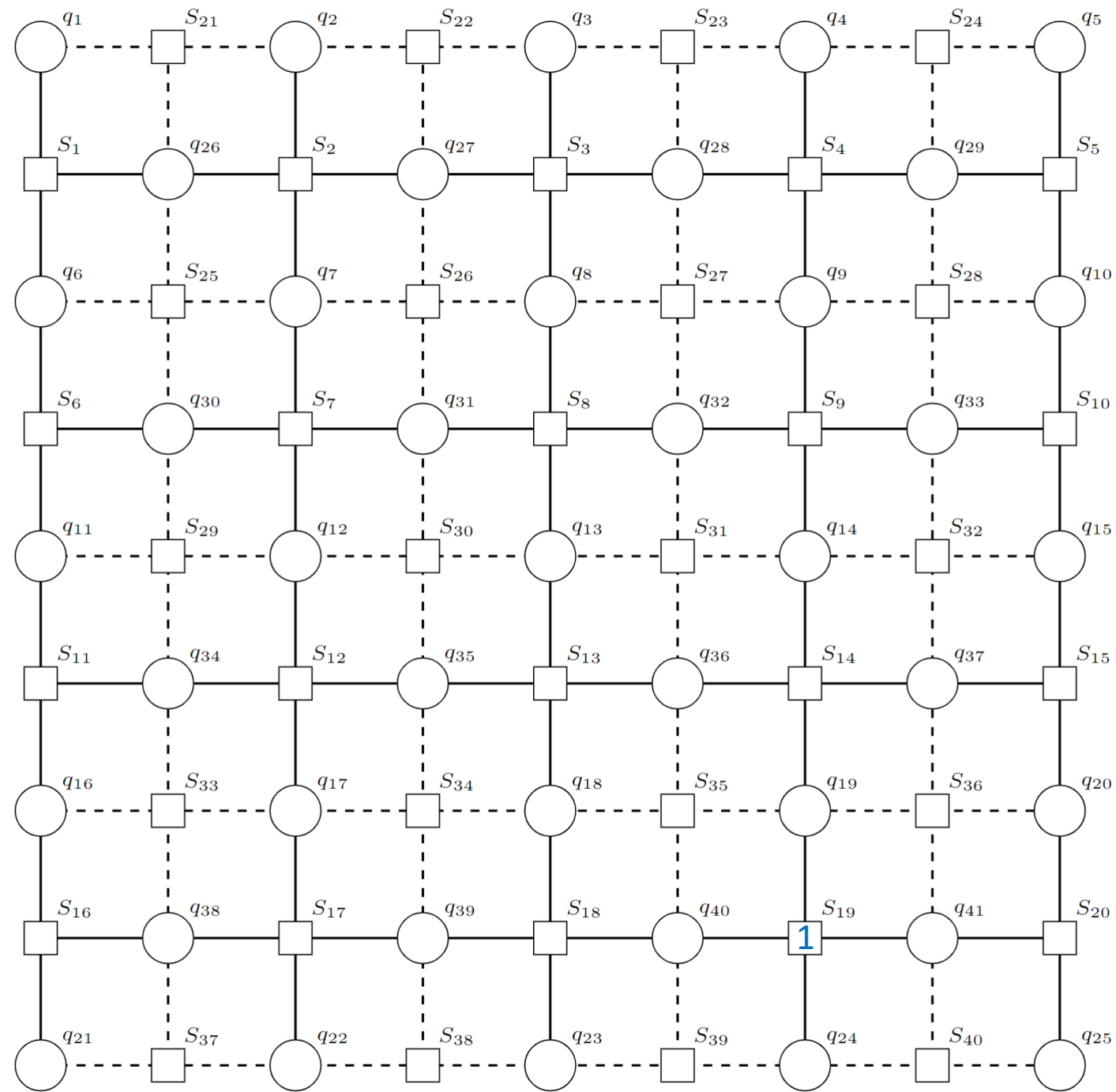


Surface codes: error detection

Surface codes are CSS codes:
X- and Z-type errors are
detected by different
stabilisers.



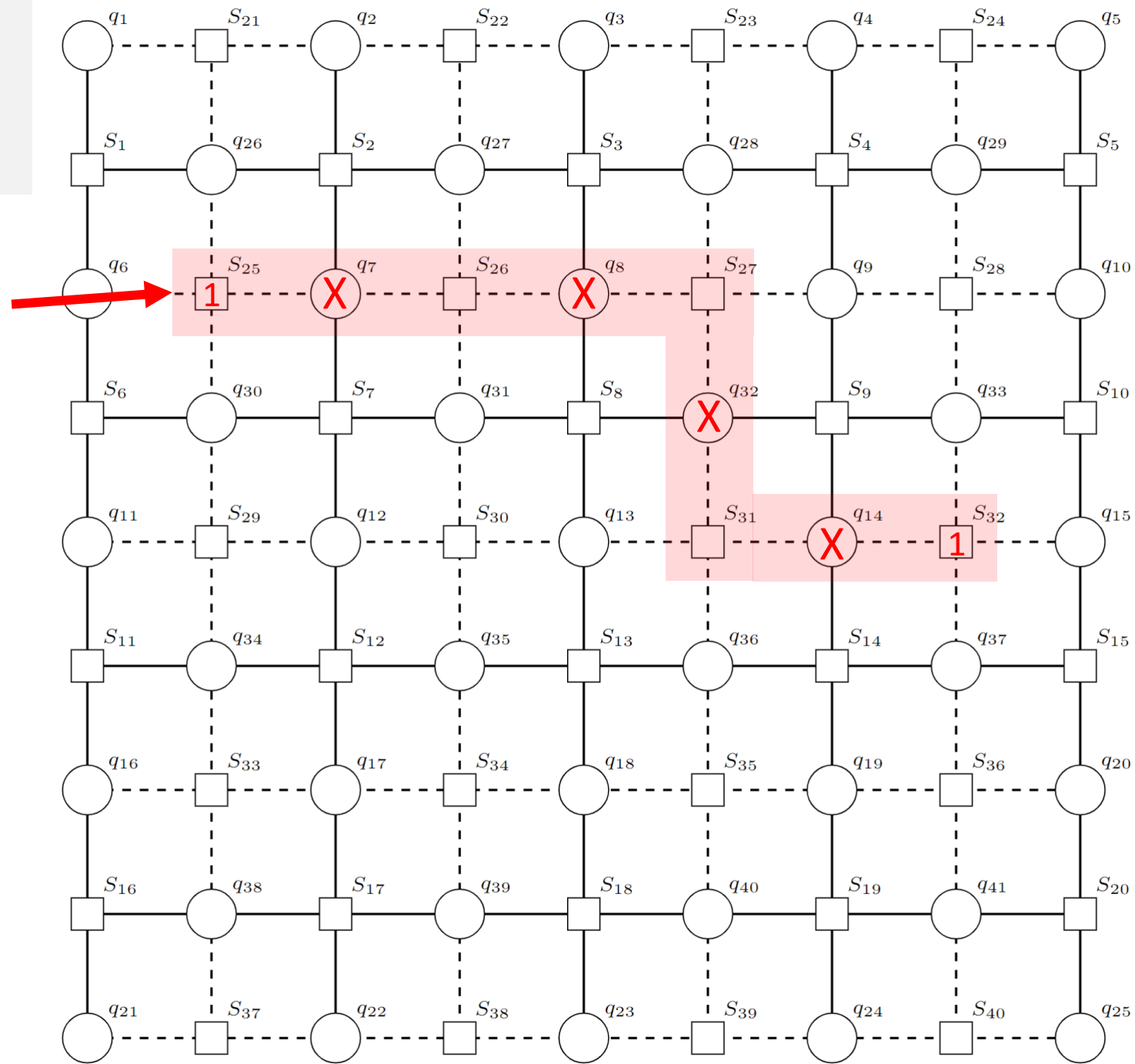
Surface code: error chains



Surface code: error chains

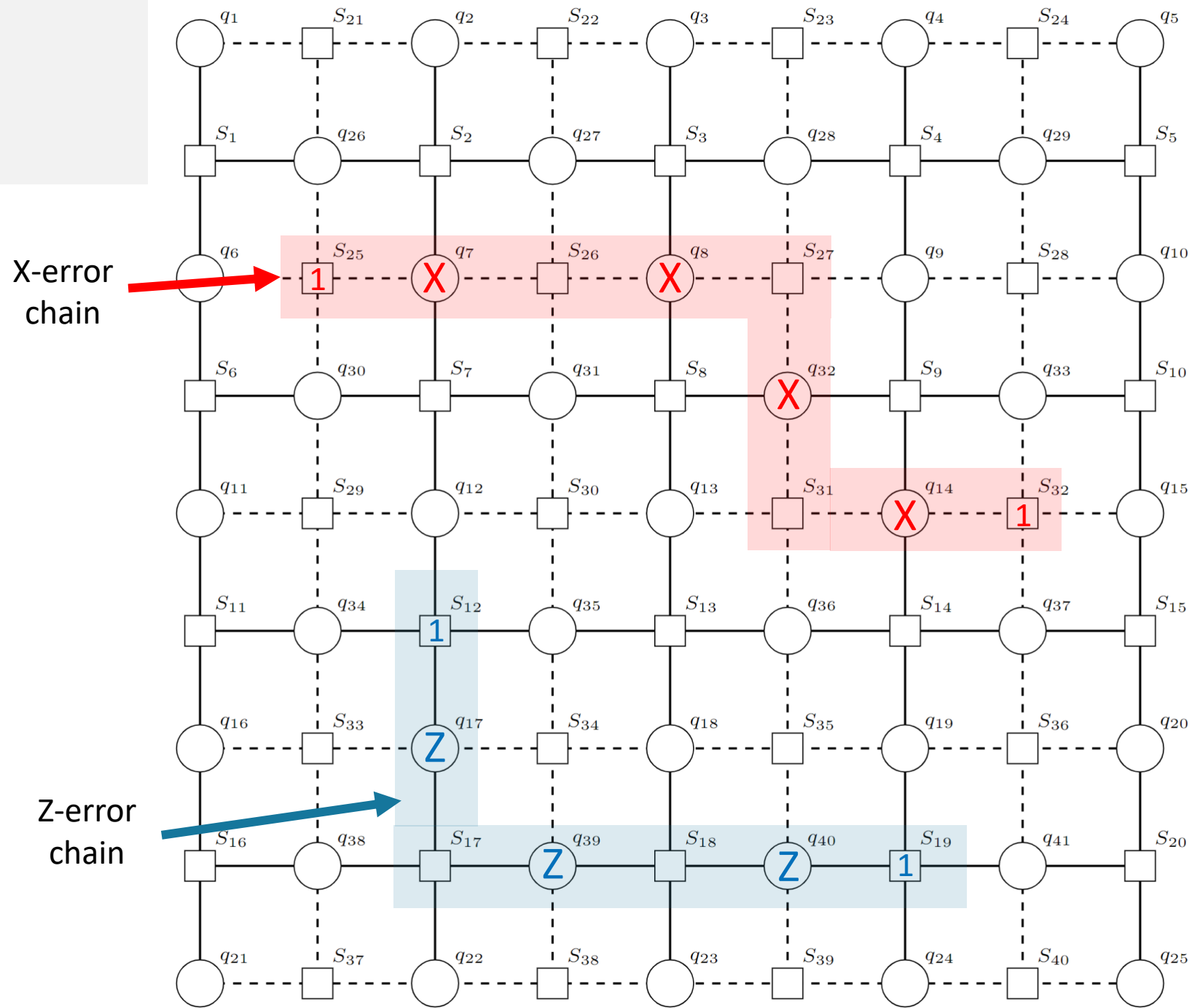
Error chains in the bulk
have syndrome
Hamming weight 2.

X-error
chain



Surface code: error chains

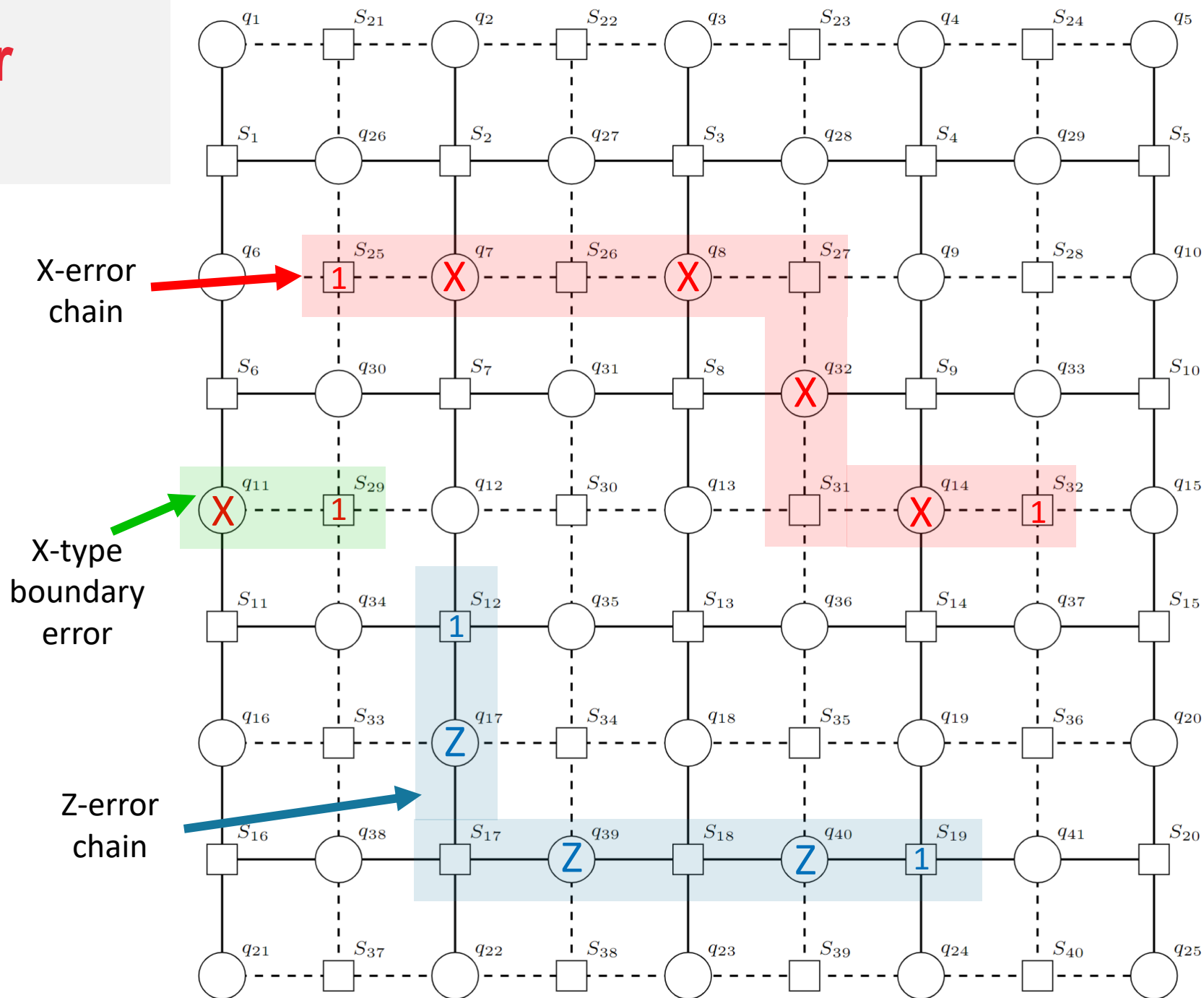
Error chains in the bulk have syndrome Hamming weight 2.



Surface code: error chains

Error chains in the bulk have syndrome Hamming weight 2.

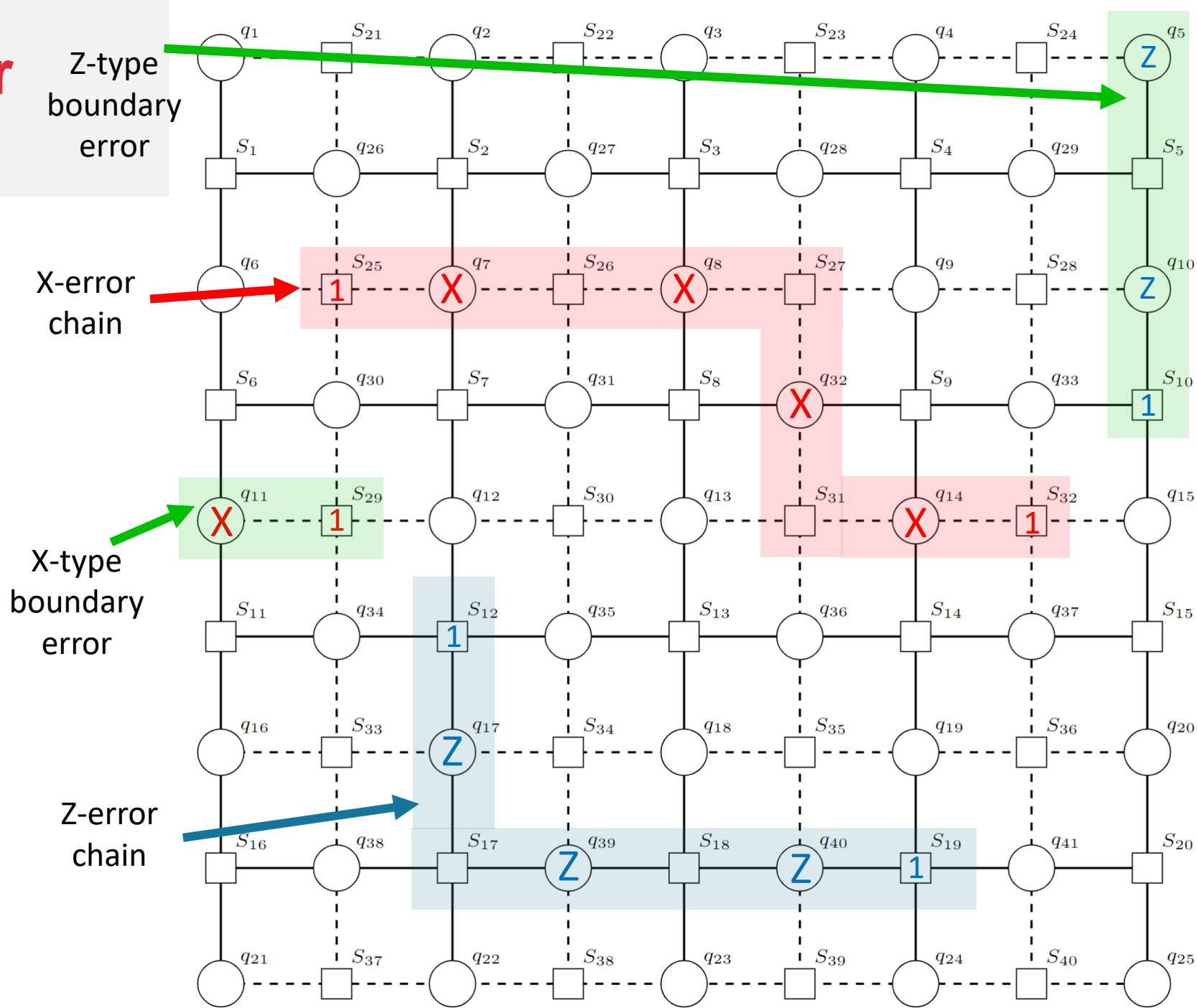
Error chains starting at the boundary have syndrome Hamming weight 1.



Surface code: error chains

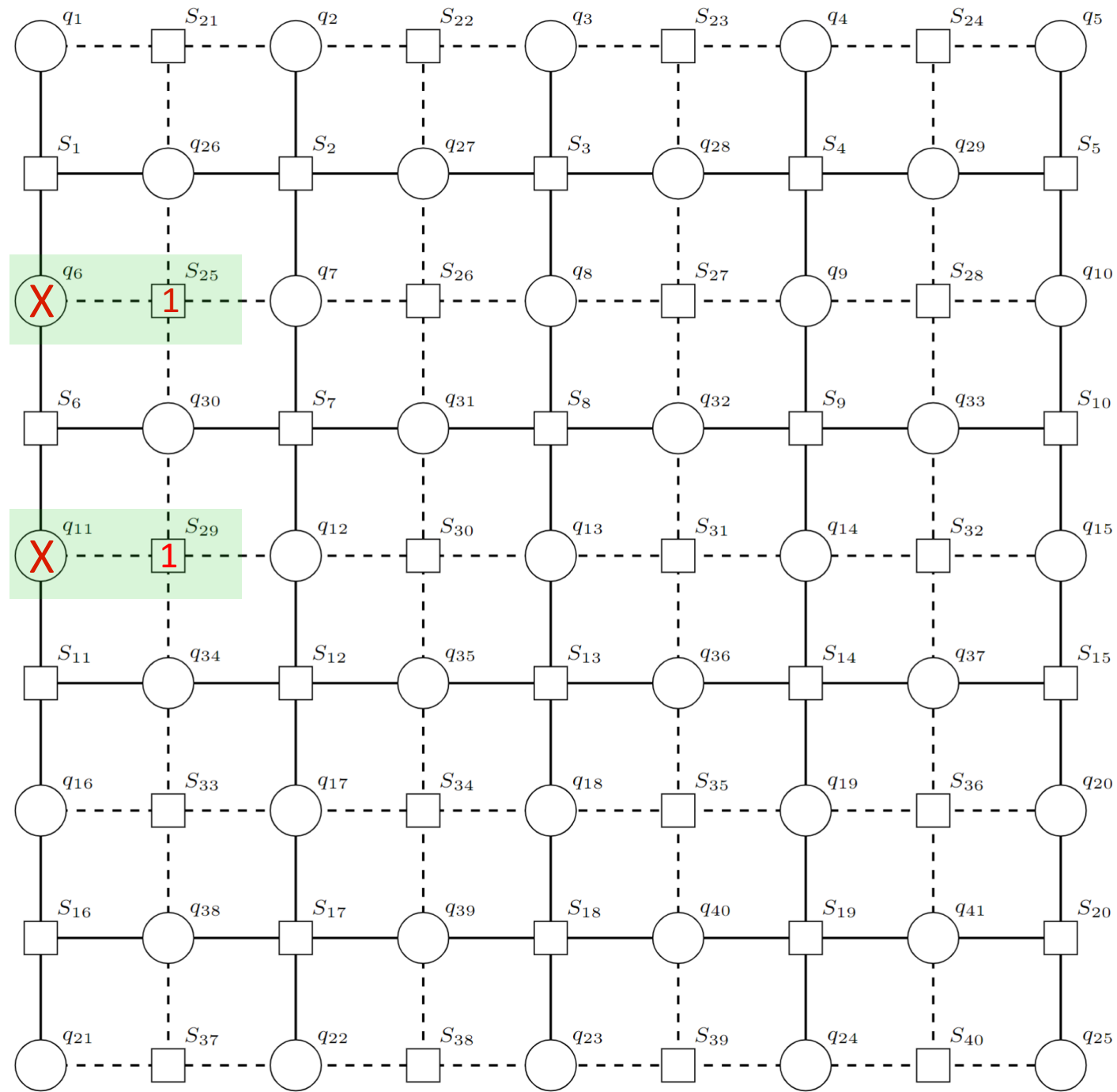
Error chains in the bulk have syndrome Hamming weight 2.

Error chains starting at the boundary have syndrome Hamming weight 1.



Zero-syndrome error chains

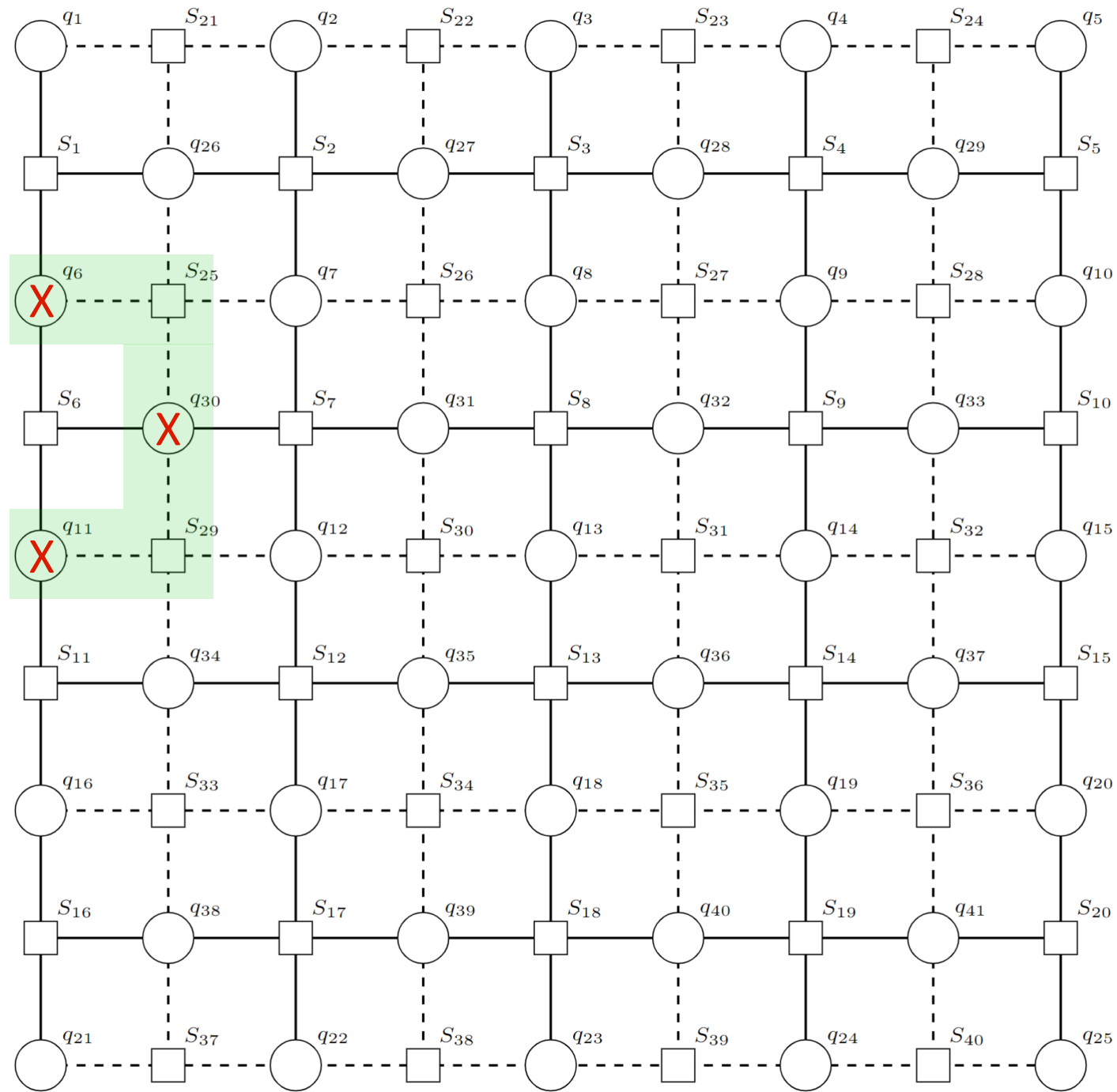
If two boundary error chains merge, we get an **undetectable** zero-weight syndrome.



Zero-syndrome error chains

If two boundary error chains merge, we get an **undetectable** zero-weight syndrome.

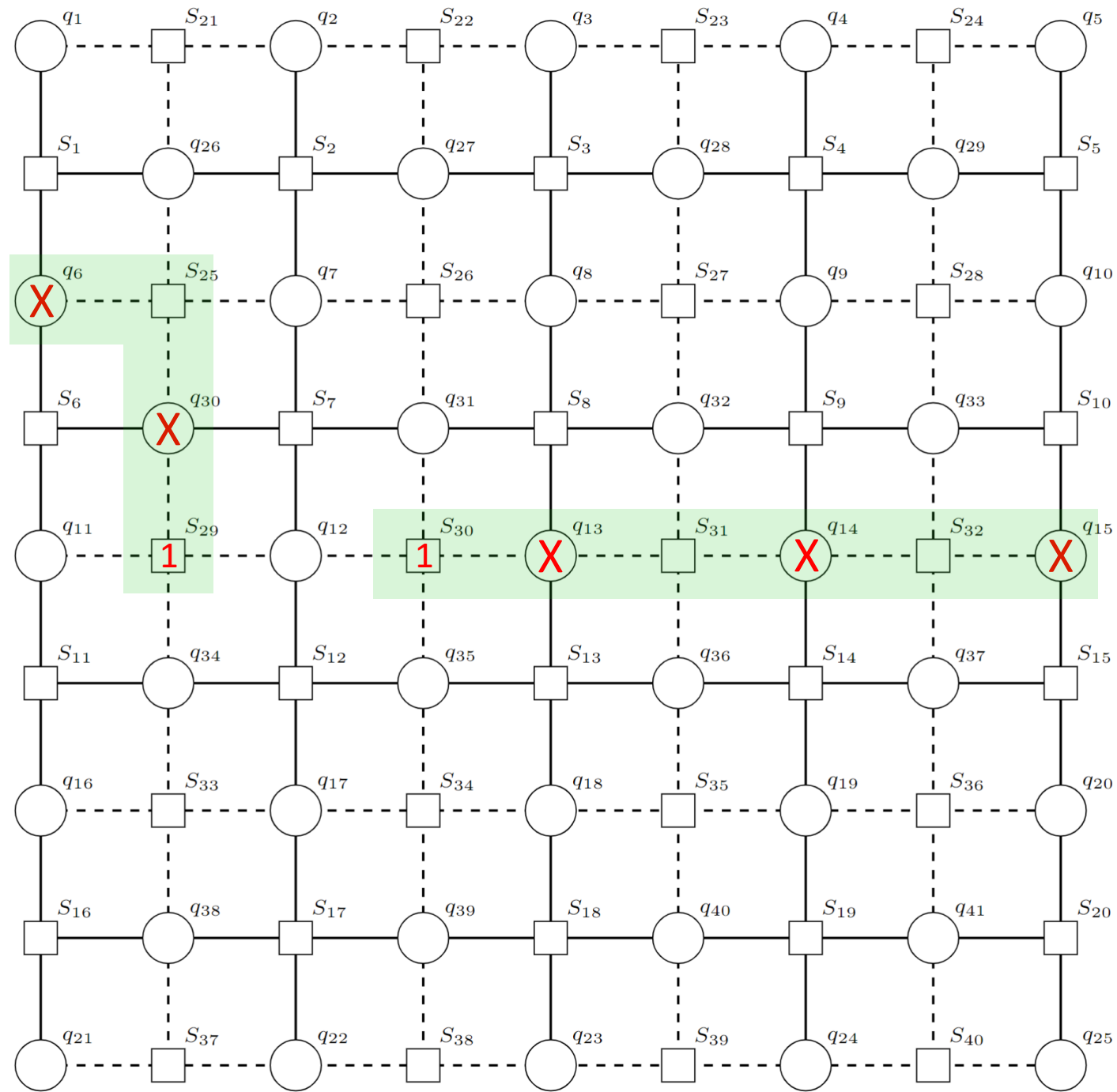
The highlighted error chain is equivalent to stabiliser S_6 . \Rightarrow It has trivial action on the codespace.



Logical operators

If two boundary error chains merge, we get an **undetectable** zero-weight syndrome.

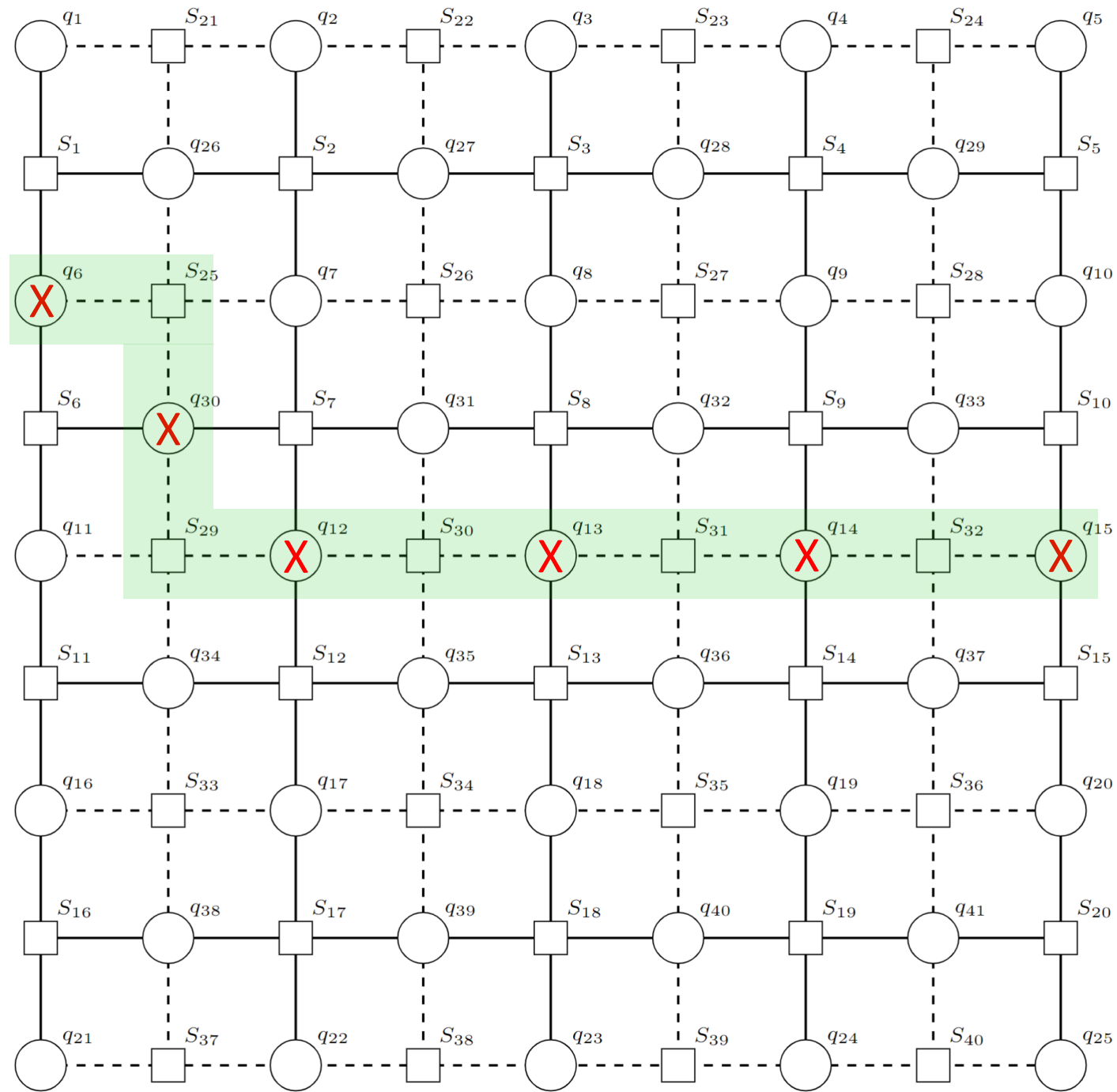
If boundary operators from opposite ends of the lattice merge, we get a non-zero syndrome error chain that is not in the stabiliser group. This is a **logical operator**.



Logical operators

If two boundary error chains merge, we get an **undetectable** zero-weight syndrome.

If boundary operators from opposite ends of the lattice merge, we get a non-zero syndrome error chain that is not in the stabiliser group. This is a **logical operator**.



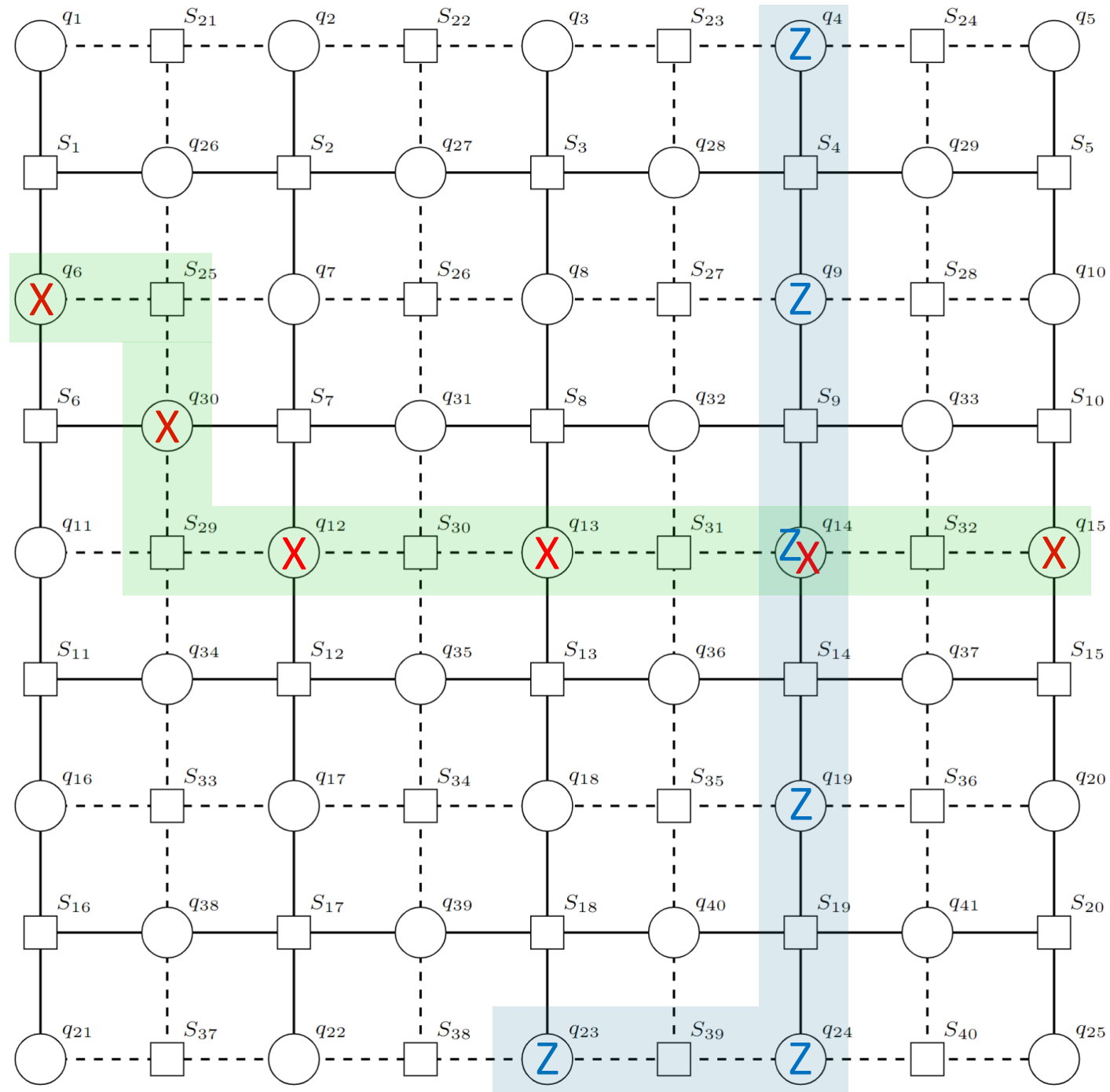
X- and Z-logical operators

X-Logical operators span from left-to-right boundary.

Z-Logical operators span from top-to-bottom boundary

We also see that the two logical operator anti-commute as they intersect on one qubit:

$$\{X_L, Z_L\} = 0$$



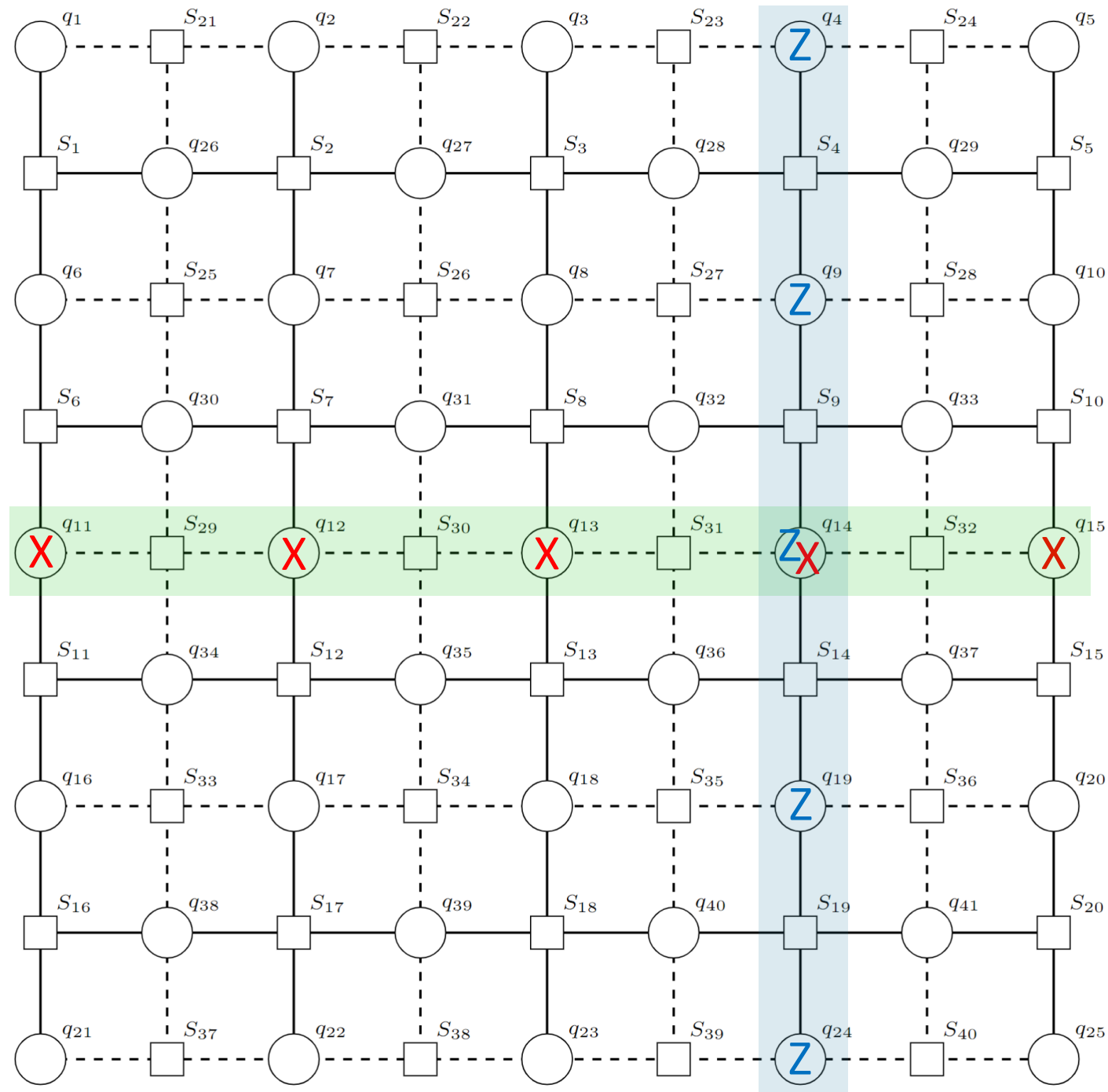
Surface code distance

The code distance corresponds to the minimum-weight logical operator.

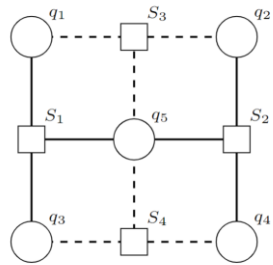
In the surface code, these operators are error chains that span directly from one boundary to another.

The distance of the surface code is given by the size of the lattice.

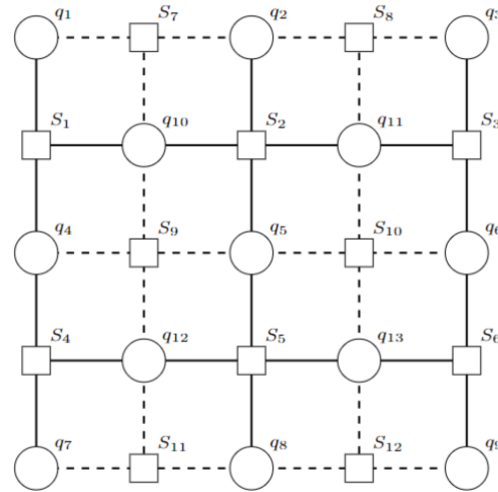
This surface code is defined on a 5x5 lattice. The distance $d = 5$.



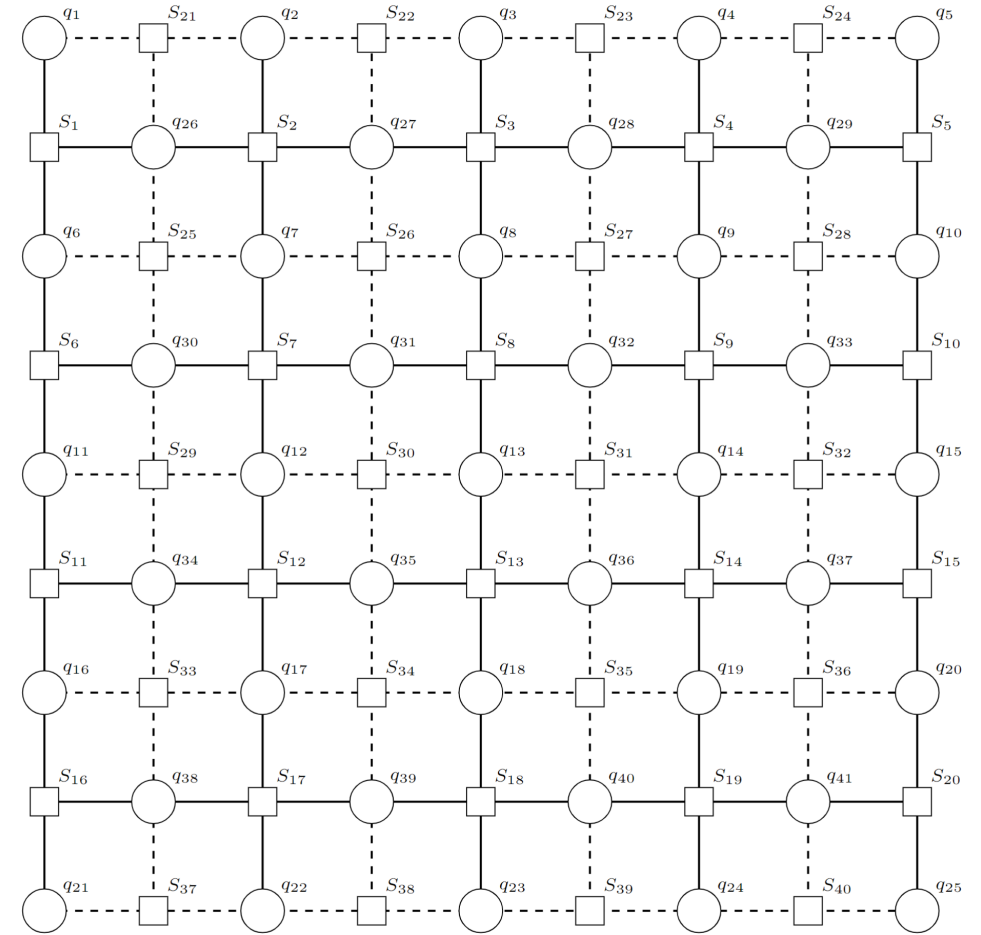
Scaling the Surface Code



$d = 2$ Surface code



$d = 3$ Surface code



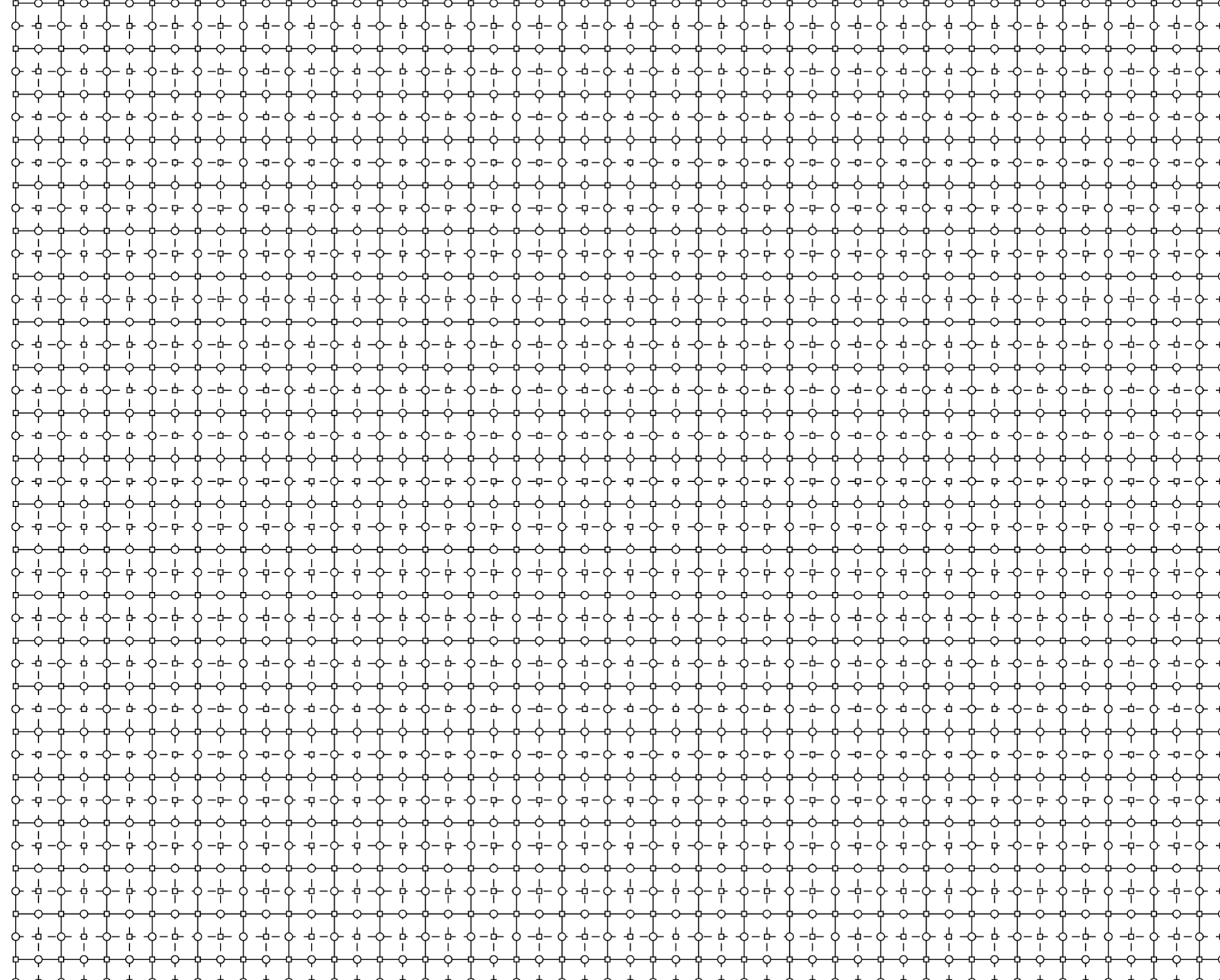
$d = 5$ Surface code

The distance of surface code scales $d \sim O(\sqrt{n})$, where n is the number of physical qubits. By increasing the distance we can arbitrarily suppress the logical error rate.

Q: What sort of distances will we need to suppress the logical error rate to algorithmically useful levels $p_L \approx 10^{-12}$


A: Estimates suggest $d \approx 27$ will be required. This corresponds to a surface code patch with > 1000 physical qubits.

Over 99.9% of the qubits in a quantum computer will be dedicated to error correction!



When will surface code quantum computers be useful?

- The Google Team have implemented surface codes on a 107 qubit chip.
- Estimates suggest operationally useful quantum computers will require surface codes of size $d \approx 27$ on chips >1000 qubits.

 > quant-ph > arXiv:1905.09749

Search...
Help | Advan

Quantum Physics

[Submitted on 23 May 2019 (v1), last revised 13 Apr 2021 (this version, v3)]

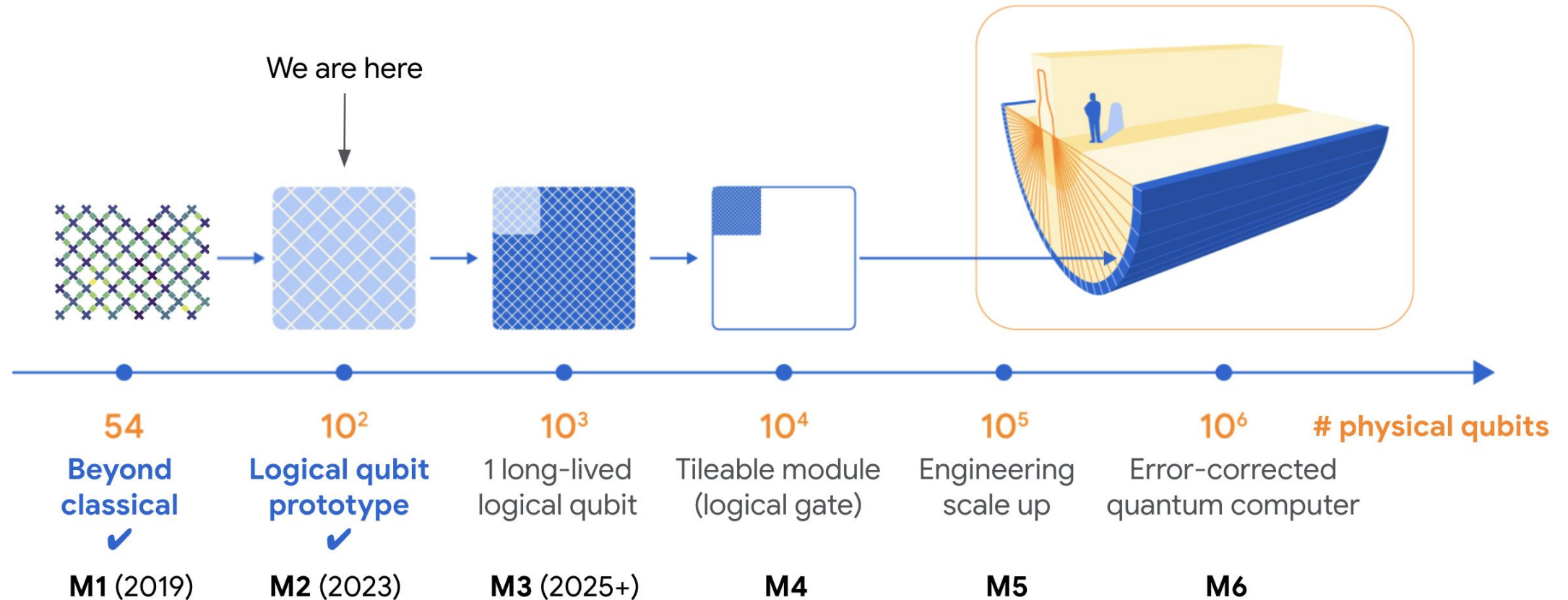
How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits

Craig Gidney, Martin Ekerå

We significantly reduce the cost of factoring integers and computing discrete logarithms in finite fields on a quantum computer by combining techniques from Shor 1994, Griffiths-Niu 1996, Zalka 2006, Fowler 2012, Ekerå-Håstad 2017, Ekerå 2017, Ekerå 2018, Gidney-Fowler 2019, Gidney 2019. We estimate the approximate cost of our construction using plausible physical assumptions for large-scale superconducting qubit platforms: a planar grid of qubits with nearest-neighbor connectivity, a characteristic physical gate error rate of 10^{-3} , a surface code cycle time of 1 microsecond, and a reaction time of 10 microseconds. We account for factors that are normally ignored such as noise, the need to make repeated attempts, and the spacetime layout of the computation. When factoring 2048 bit RSA integers, our construction's spacetime volume is a hundredfold less than comparable estimates from earlier works (Van Meter et al. 2009, Jones et al. 2010, Fowler et al. 2012, Gheorghiu et al. 2019). In the abstract circuit model (which ignores overheads from distillation, routing, and error correction) our construction uses $3n + 0.002n \lg n$ logical qubits, $0.3n^3 + 0.0005n^3 \lg n$ Toffolis, and $500n^2 + n^2 \lg n$ measurement depth to factor n -bit RSA integers. We quantify the cryptographic implications of our work, both for RSA and for schemes based on the DLP in finite fields.



Google's Roadmap to Fault-Tolerance





THE UNIVERSITY *of* EDINBURGH
informatics

www.informatics.ed.ac.uk