Almost-linear time decoding algorithm for topological codes

Naomi Nickerson & Nicolas Delfosse

Delfosse & Nickerson 1709.06218
Delfosse & Zemor 1703.01517

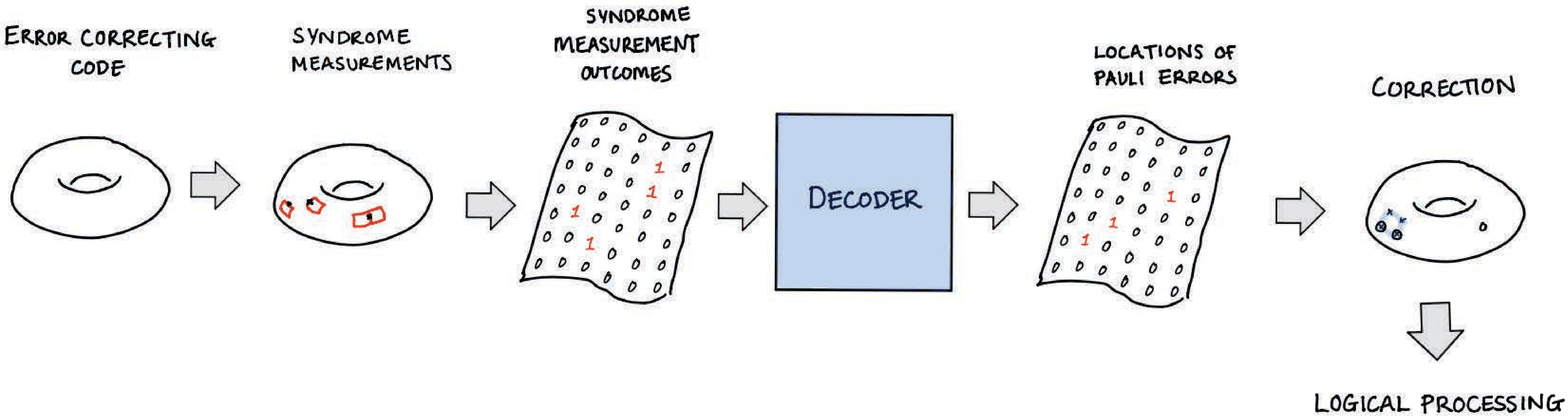
QIP, 17th January 2018

1. What is decoding and how fast does it need to be?

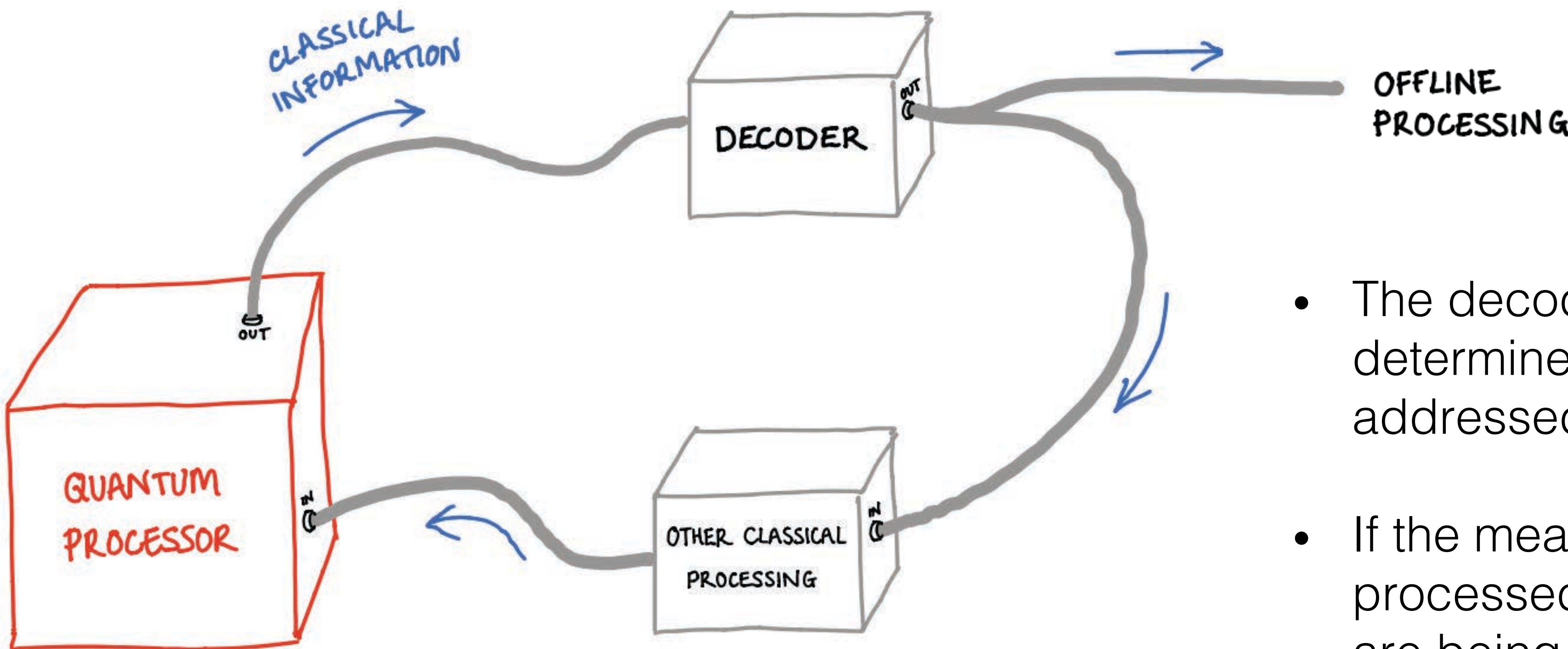
2. The Union Find decoder

3. Achieving almost-linear time

What is a decoder?



How fast?

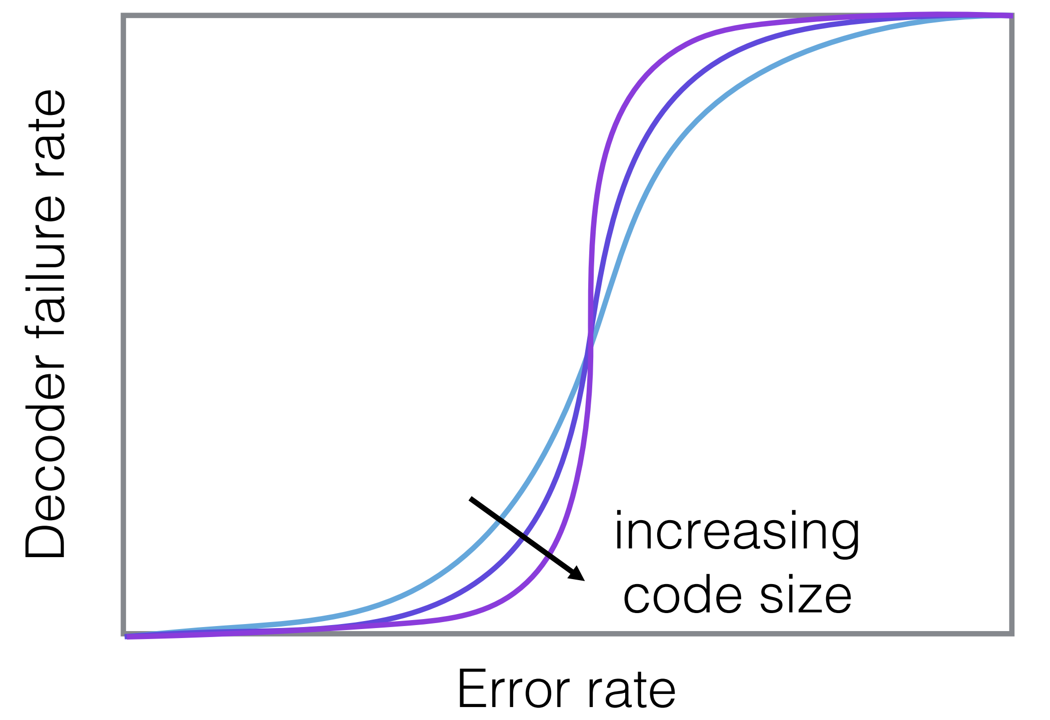


- The decoder is part of a feedback loop that determines how the quantum state should be addressed.
- If the measurements outcomes cannot be processed by the decoder as quickly as they are being produced, we end up with a backlog problem.

	Clock speed	Estimated decoding time (for a 20x20 lattice)	
Superconducting qubits	**20ns**	**400 ns**	1411.7403
Trapped ions	**480ns**	**10,000 ns**	1709.06952
Photons	**2 ns**	**40 ns**	
NV centres (electron)	**10 μs**	**200,000 ns**	Science 356, 634, 928-932
NV centres (nuclear spin)	**500 μs**	**10,000,000 ns**	Science 356, 634, 928-932

What makes a good decoder?

- The optimal decoder cannot always correct errors, instead we see **threshold behavior**
- The optimal decoder has exponential complexity. Approximate algorithms trade off speed for lowering the threshold.



	Threshold (2d surface code)	Worst case Complexity	**Works with any geometry?**
Optimal decoder	11.0%	e^N	✔
MWPM (Harrington, Fowler)	10.3%	$O(N^3)$	x
RG decoder (Harrington, Duclos-Cianci, Poulin)	8.2%	O(NlogN)	x
HDRG (Wooton)	7.3%	$O(N^2)$	✔
Union Find decoder	**9.9%**	**$O(\alpha(N)\ N)$**	✔

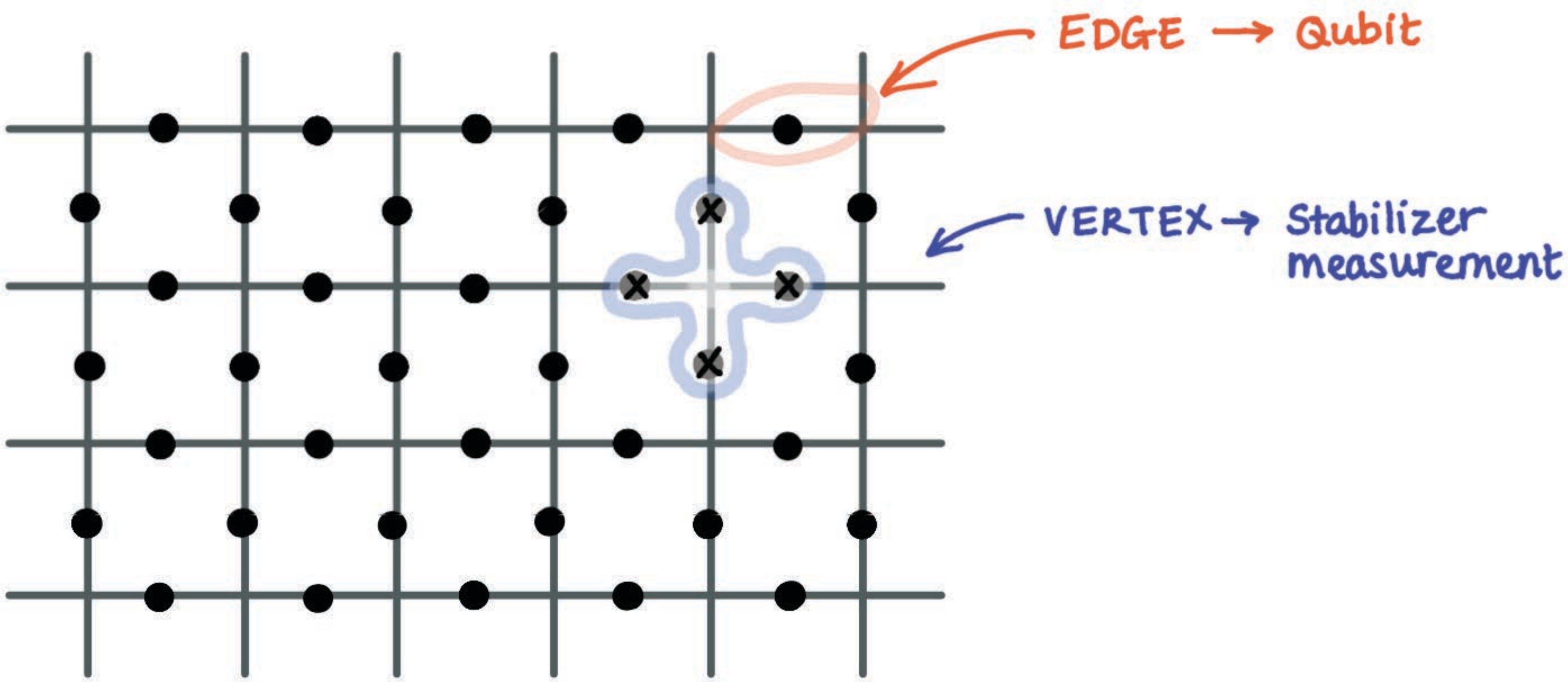
1. What is decoding and how fast does it need to be?

2. The Union Find decoder

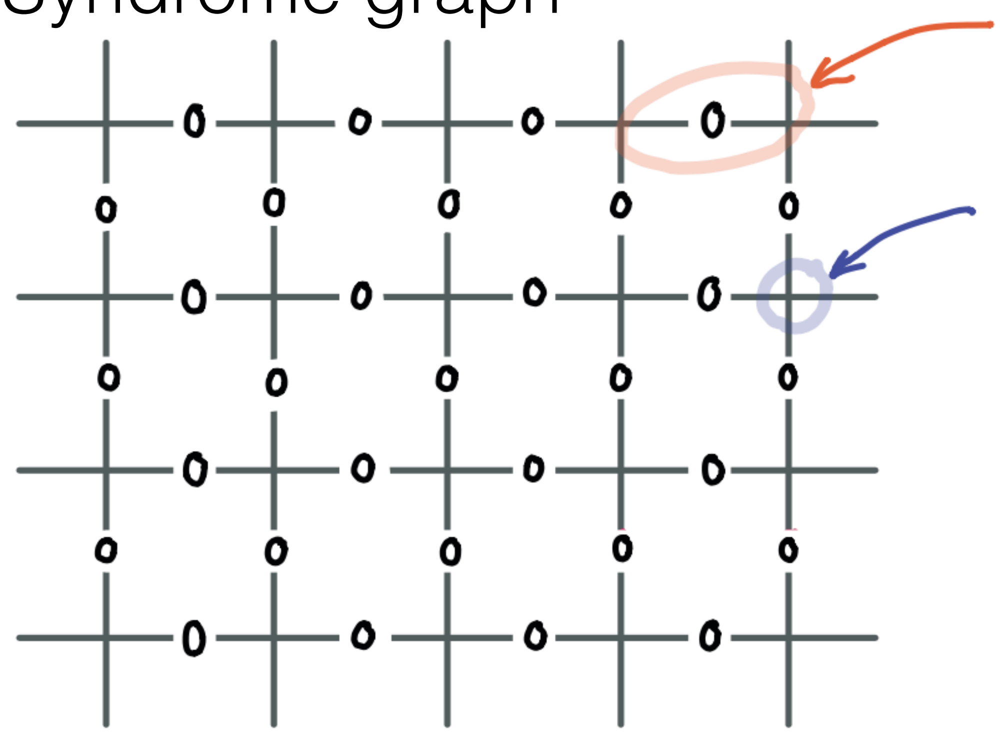
3. Achieving almost-linear time

Syndrome graph

2D surface code



Syndrome graph



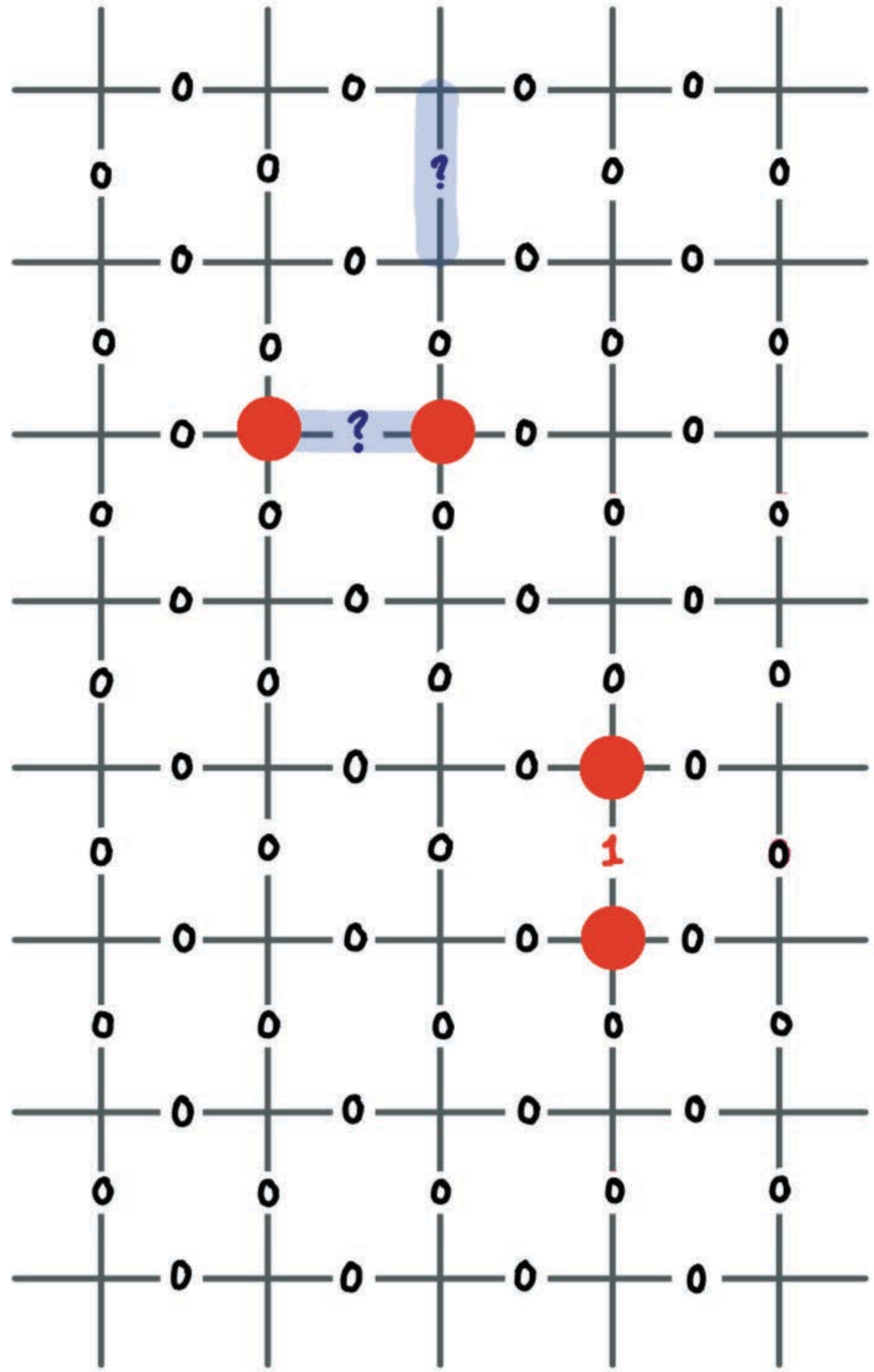
- The syndrome graph is a representation of (one basis of) the code.
- Each edge represents the error state of a qubit.
- Stabilizer measurement outcomes are associated with each vertex.
- The sum of the outcomes around each vertex should always be even.

Identifying errors

Erasure error

a.k.a loss

- A Pauli error occurs with 50% probability
- The location of the error is known.
- For example: we reinitialize a qubit into the maximially mixed state.



Bit-flip error

(Z-basis Pauli error)

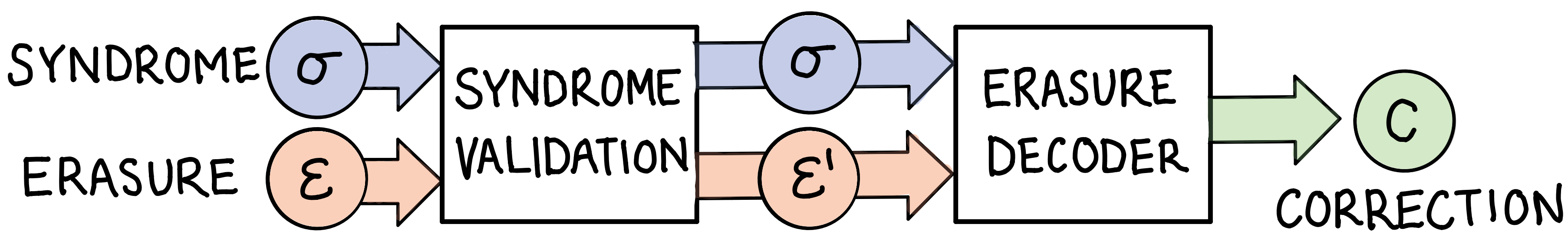
- The location of the error is unknown
- For example: the photon is in the 'wrong' rail before the final detection

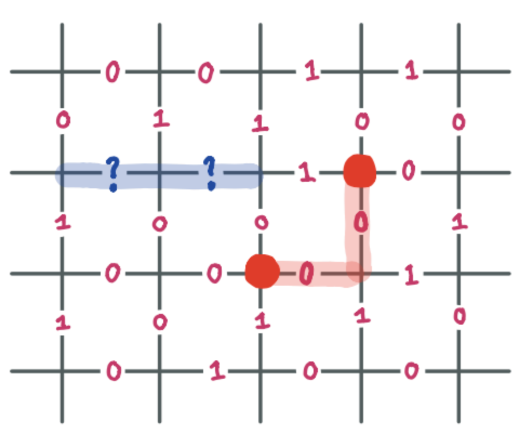
The Union Find Decoder

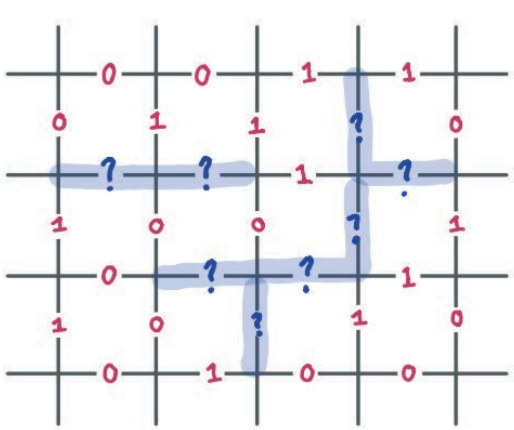
The Union Find decoding algorithm can be broken in two stages

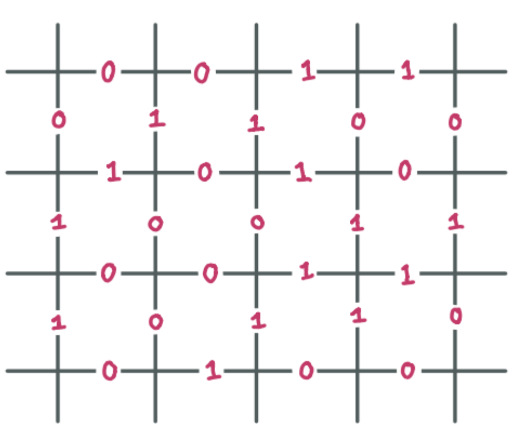
1. Convert stochastic errors into erasures

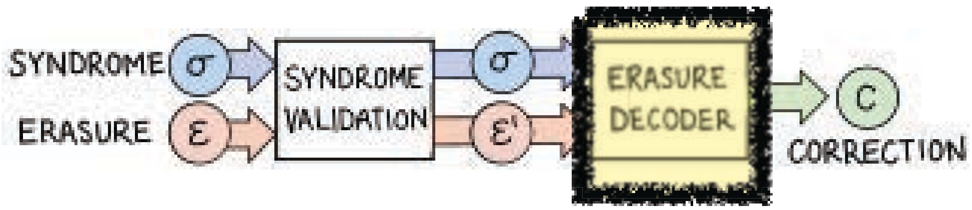
2. Apply the erasure decoder











Decoding Erasure

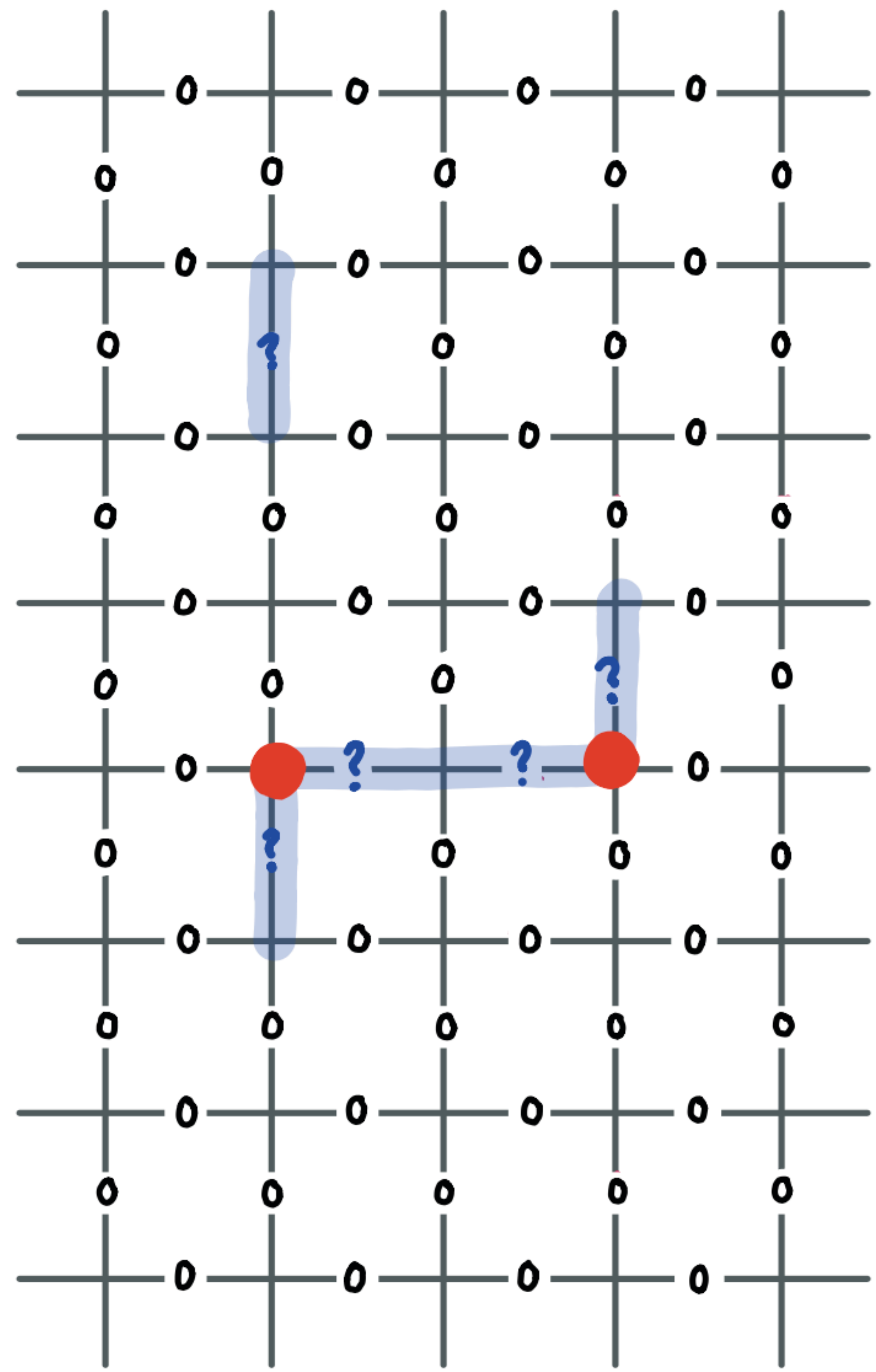
Measurement outcomes are represented as a graph where edges correspond to erased outcomes, and vertices are syndromes.

Erasure Decoder

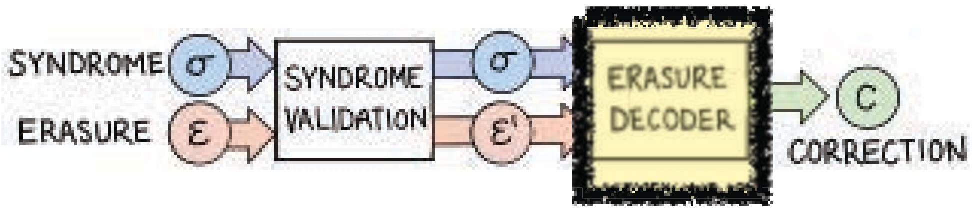
Grow a spanning forest to cover the erased edges.

while erased edges remain, do:

1. start at a leaf edge (u,v).
2. Remove the edge from the erasure, and apply the rules:
   - (R1) If the vertex u is odd, set the edge value to 1 and flip the value of v
   - (R2) If the vertex u is even, set the edge value to 0



Delfosse & Zemor 1703.01517

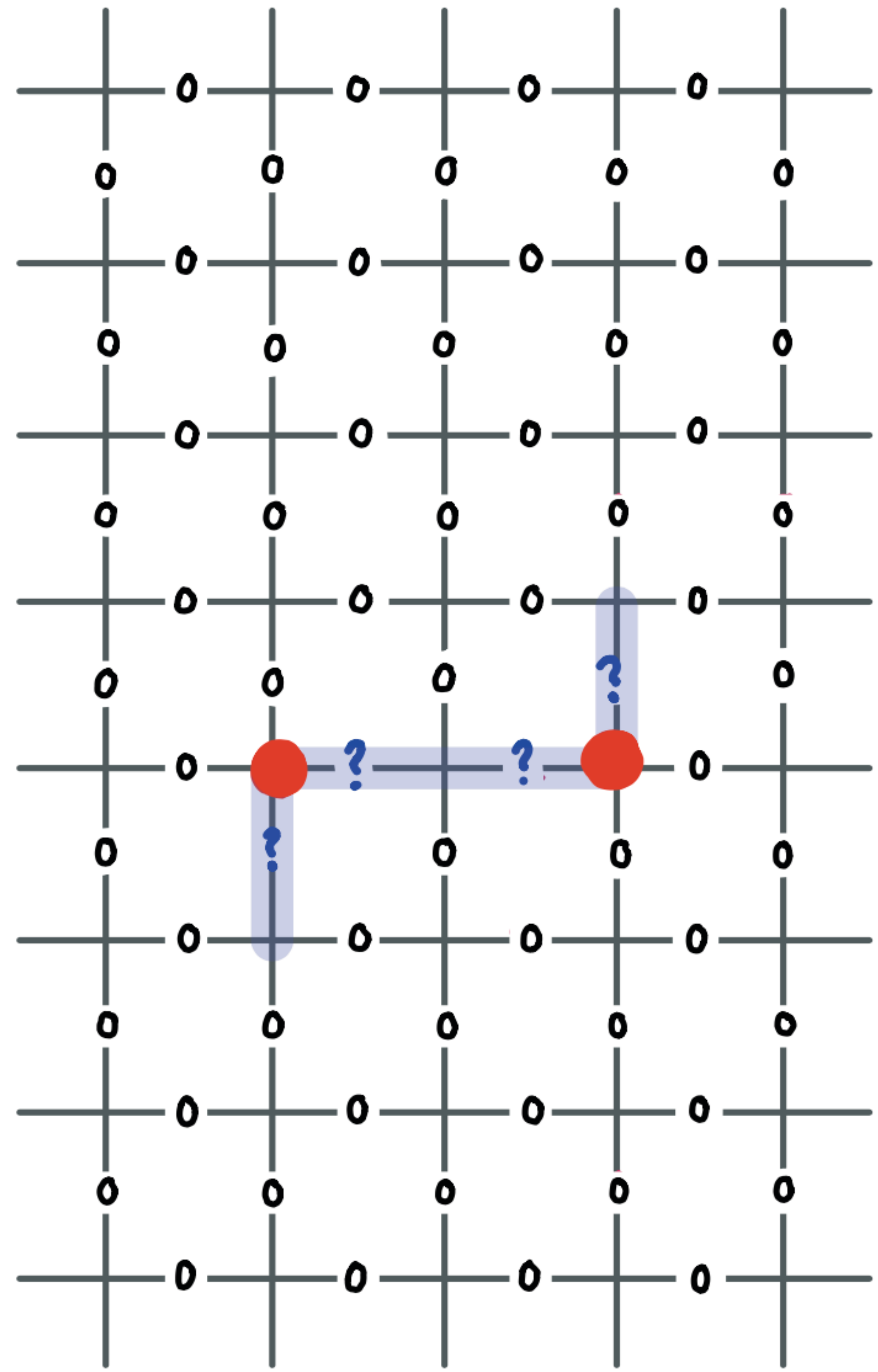


Decoding Erasure

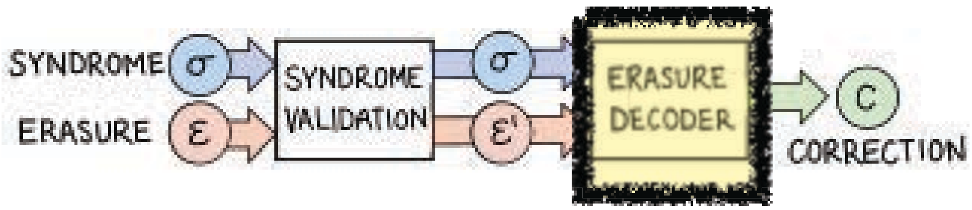
Measurement outcomes are represented as a graph where edges correspond to erased outcomes, and vertices are syndromes.

Erasure Decoder

```
Grow a spanning forest to cover the
erased edges.

while erased edges remain, do:

    1. start at a leaf edge (u,v).

    2. Remove the edge from the erasure, and apply the
       rules:

       • (R1) If the vertex u is odd, set the edge value
         to 1 and flip the value of v
       • (R2) If the vertex u is even, set the edge value
         to 0
```



Delfosse & Zemor 1703.01517



Decoding Erasure

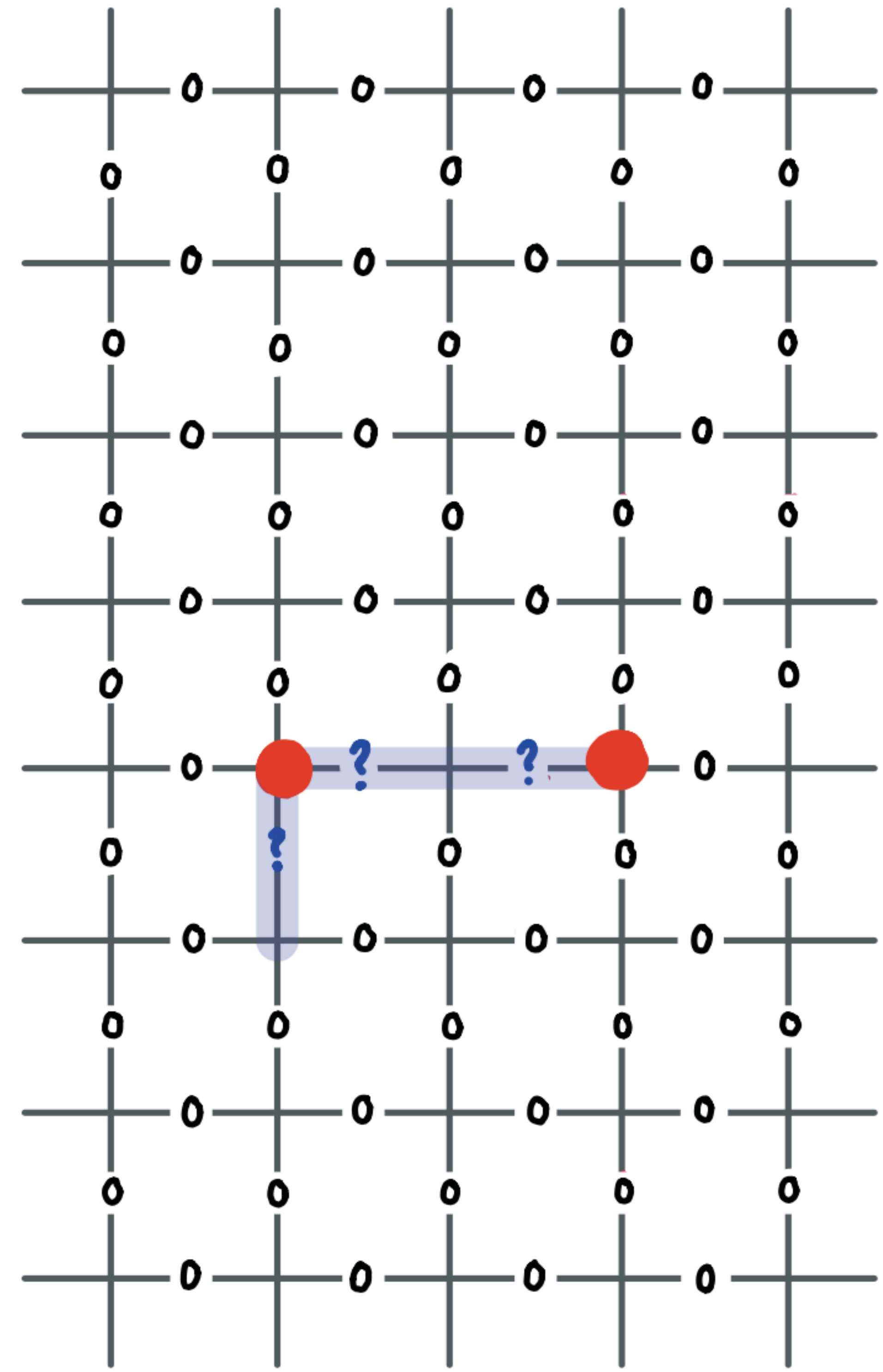
Measurement outcomes are represented as a graph where edges correspond to erased outcomes, and vertices are syndromes.

Erasure Decoder

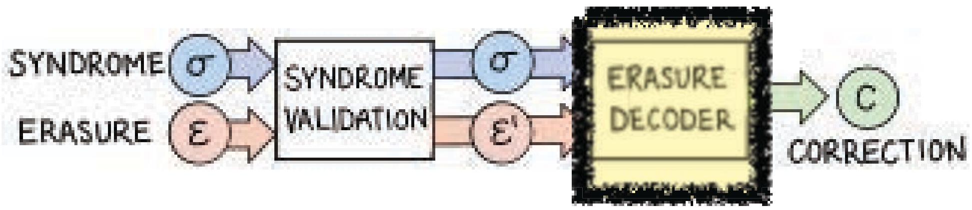
Grow a spanning forest to cover the erased edges.

while erased edges remain, do:

1. start at a leaf edge (u,v).
2. Remove the edge from the erasure, and apply the rules:
   - (R1) If the vertex u is odd, set the edge value to 1 and flip the value of v
   - (R2) If the vertex u is even, set the edge value to 0



Delfosse & Zemor 1703.01517



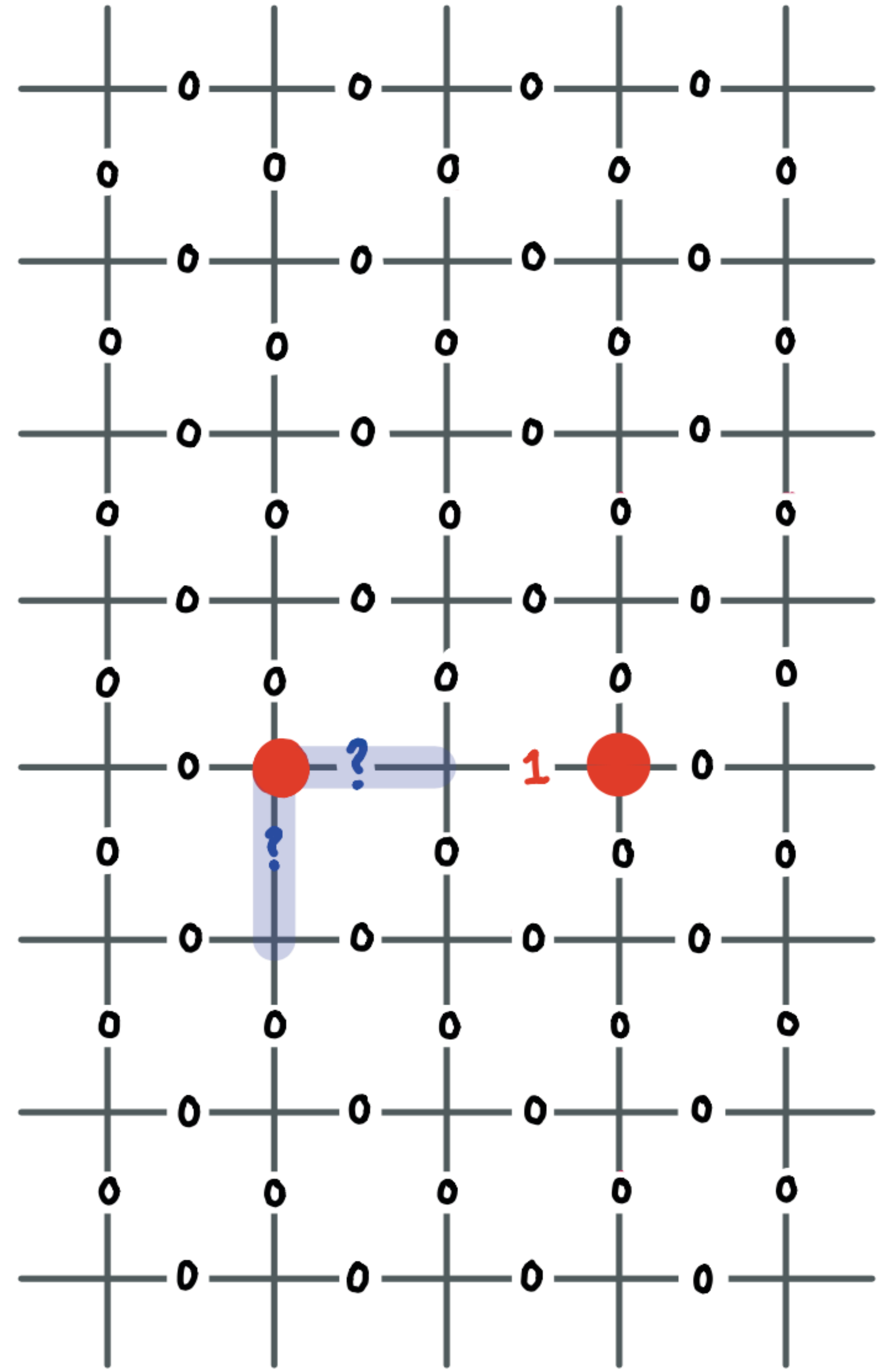
Decoding Erasure

Measurement outcomes are represented as a graph where edges correspond to erased outcomes, and vertices are syndromes.

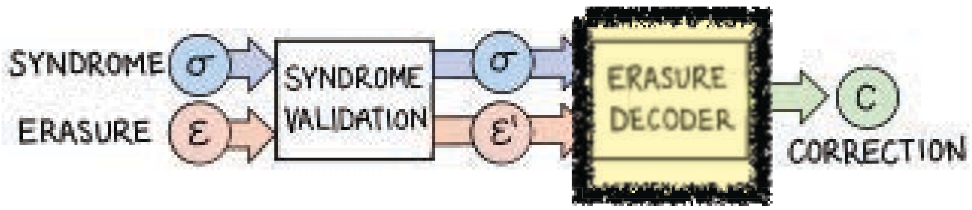
Erasure Decoder

Grow a spanning forest to cover the erased edges.

```
while erased edges remain, do:
    1. start at a leaf edge (u,v).
    2. Remove the edge from the erasure, and apply the
       rules:
        • (R1) If the vertex u is odd, set the edge value
          to 1 and flip the value of v
        • (R2) If the vertex u is even, set the edge value
          to 0
```



Delfosse & Zemor 1703.01517



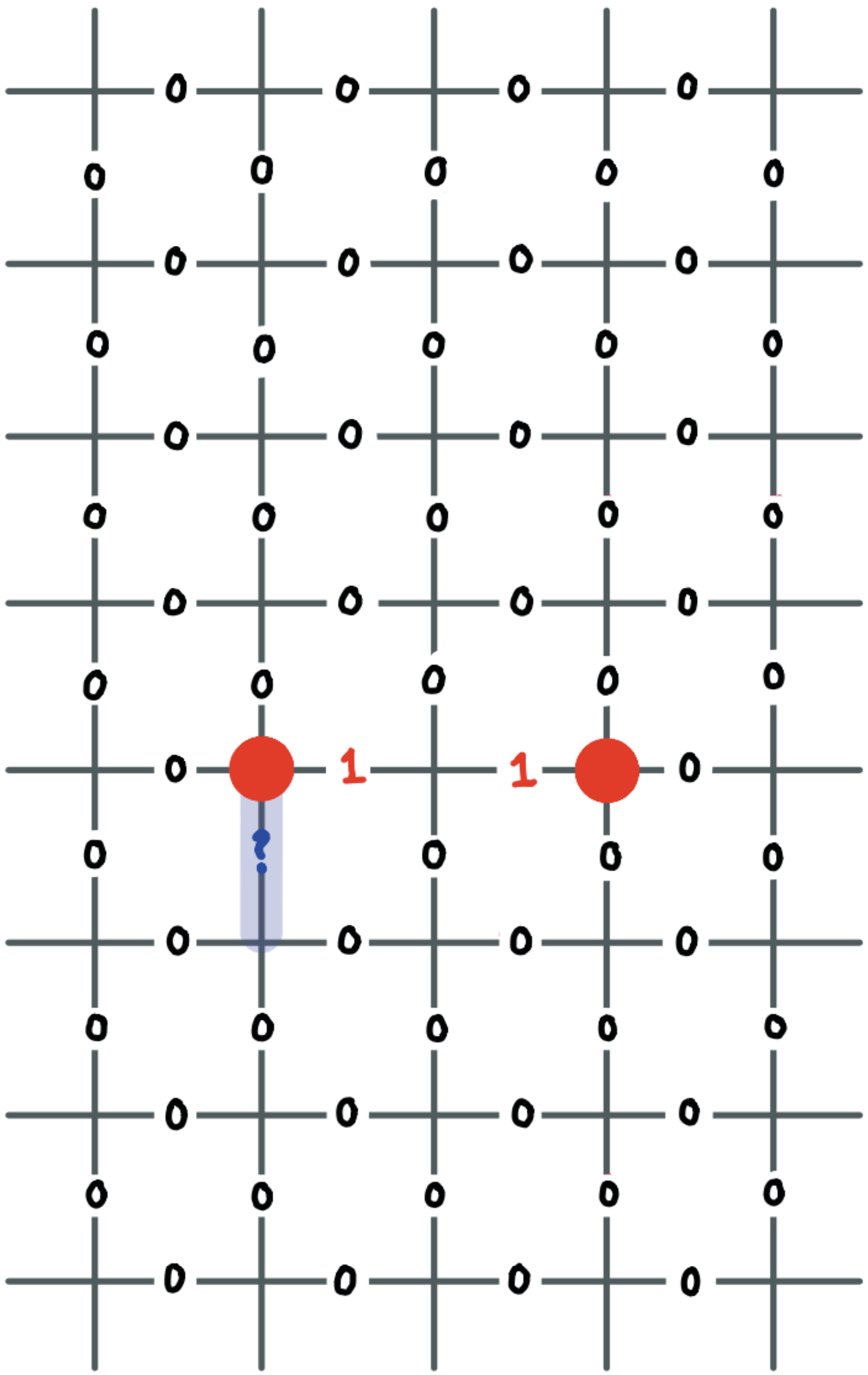
Decoding Erasure

Measurement outcomes are represented as a graph where edges correspond to erased outcomes, and vertices are syndromes.

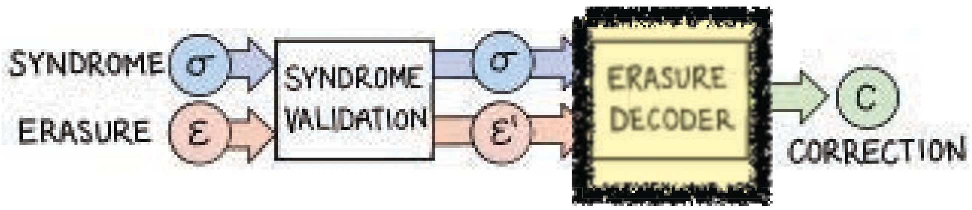
Erasure Decoder

Grow a spanning forest to cover the erased edges.

```
while erased edges remain, do:

    1. start at a leaf edge (u,v).

    2. Remove the edge from the erasure, and apply the
       rules:

       • (R1) If the vertex u is odd, set the edge value
         to 1 and flip the value of v
       • (R2) If the vertex u is even, set the edge value
         to 0
```

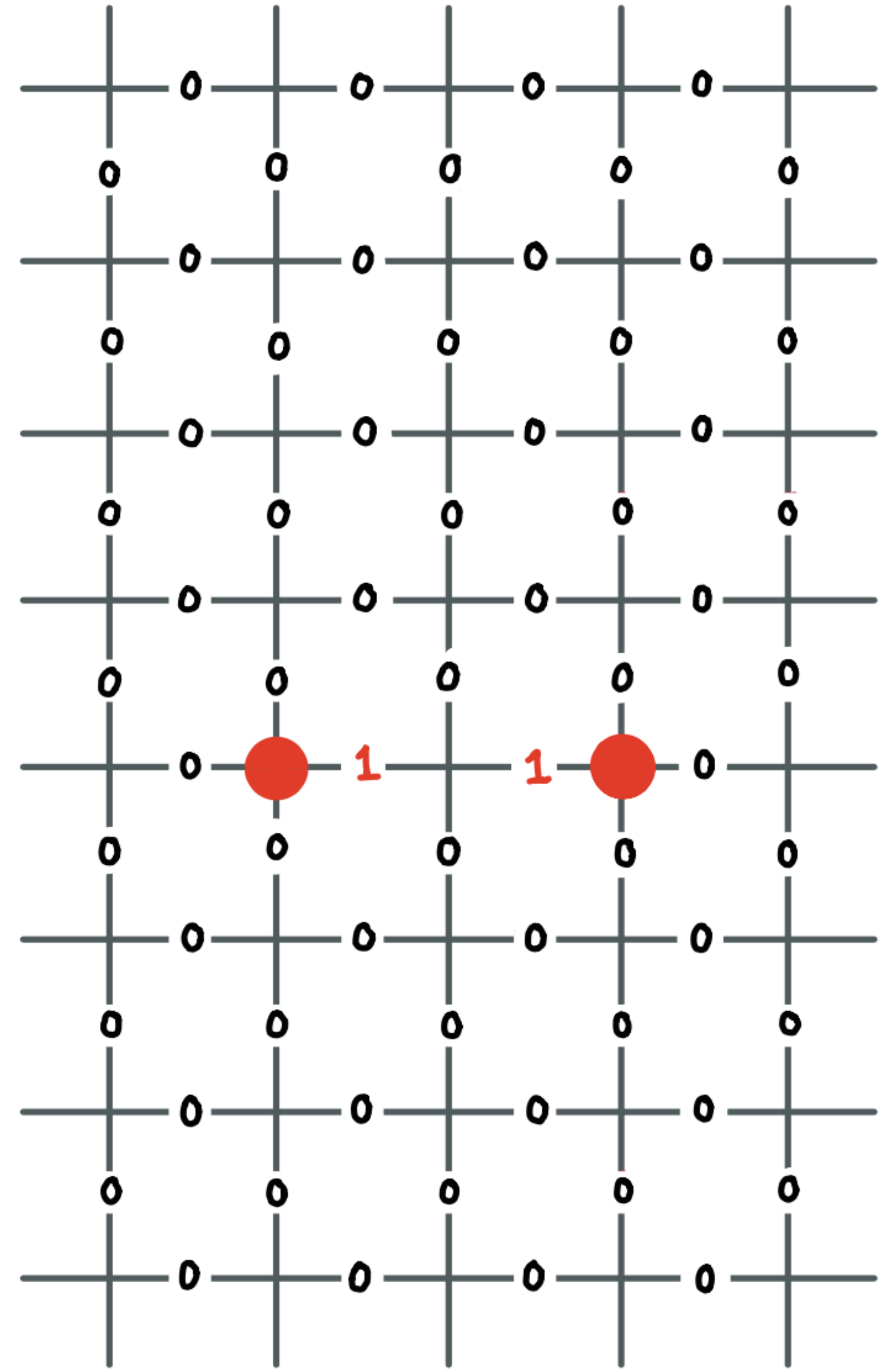


Delfosse & Zemor 1703.01517



Decoding Erasure

Measurement outcomes are represented as a graph where edges correspond to erased outcomes, and vertices are syndromes.

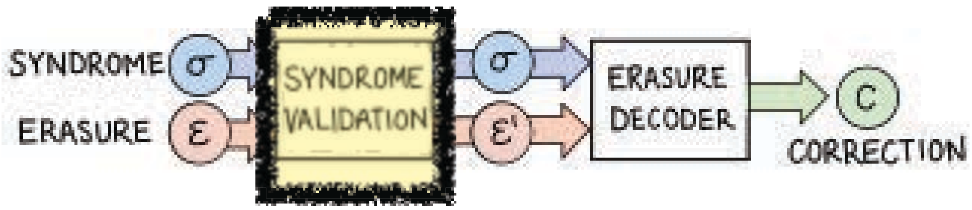


Erasure Decoder

`Grow a spanning forest to cover the erased edges.`

```
while erased edges remain, do:

    1. start at a leaf edge (u,v).

    2. Remove the edge from the erasure, and apply the
       rules:

       • (R1) If the vertex u is odd, set the edge value
         to 1 and flip the value of v
       • (R2) If the vertex u is even, set the edge value
         to 0
```

Delfosse & Zemor 1703.01517



Syndrome Validation

Syndrome Validation

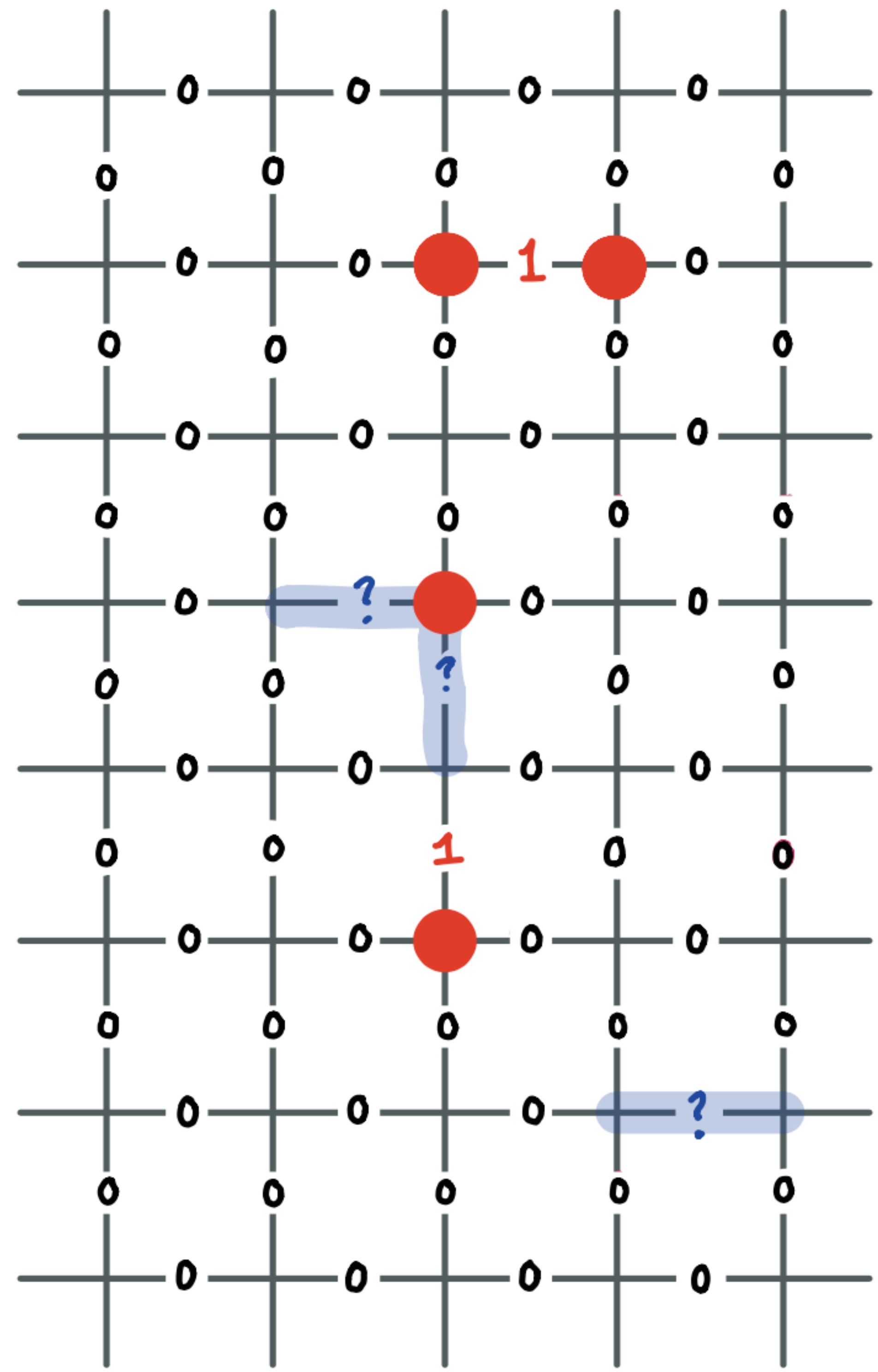
Create a list of all odd clusters

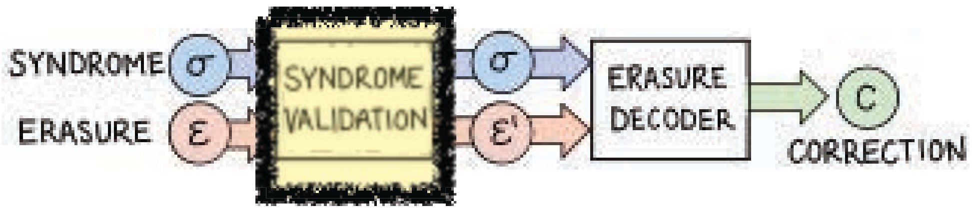
while there exists an odd cluster:

Iterate over all odd clusters:

1. Grow the cluster by a half-edge
2. If the cluster meets another cluster, fuse and update parity
3. If the new cluster is even, remove it from the odd cluster list

Add any edges that are full grown to the list of erased edges.





Syndrome Validation

Syndrome Validation

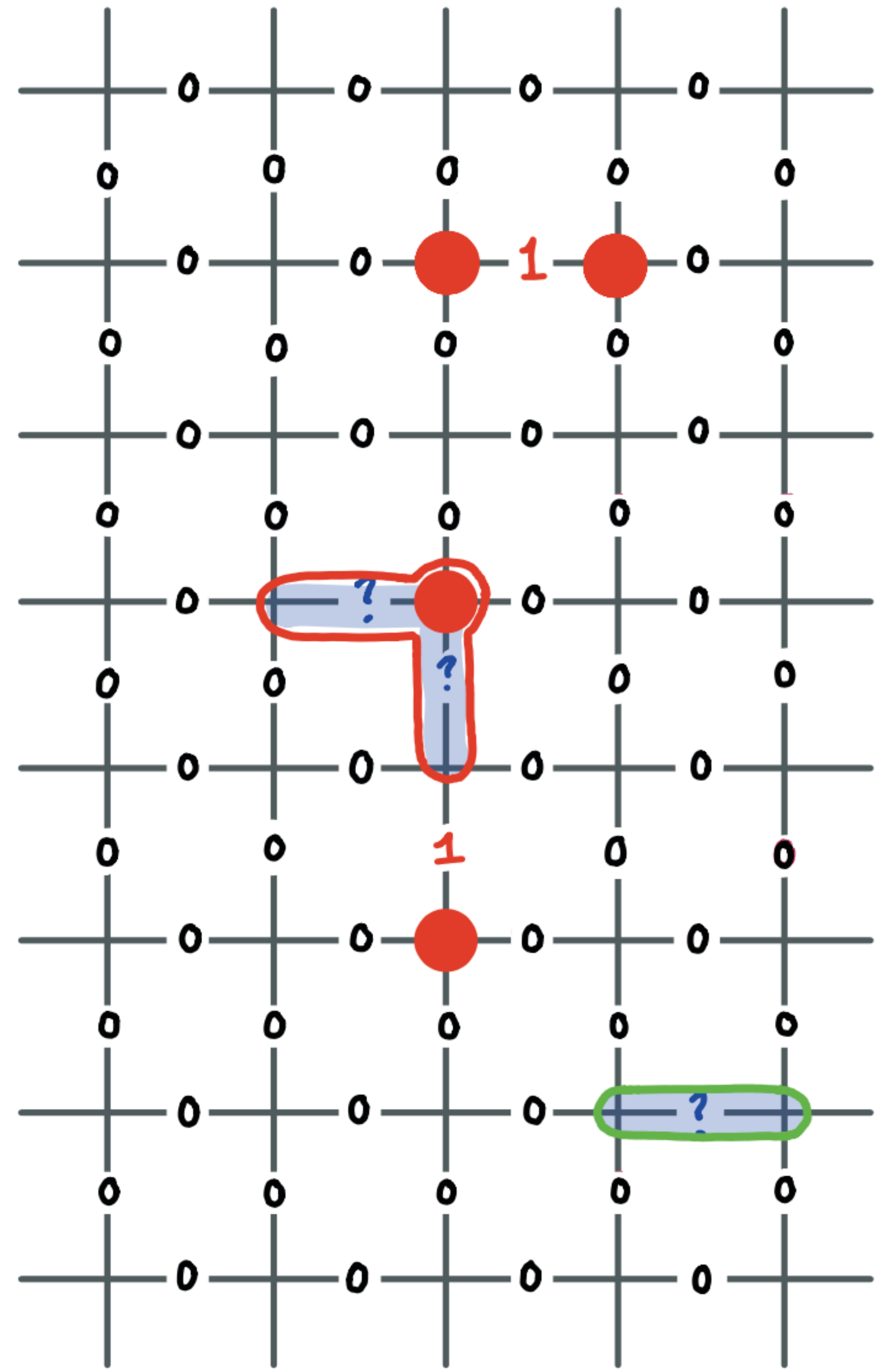
Create a list of all odd clusters

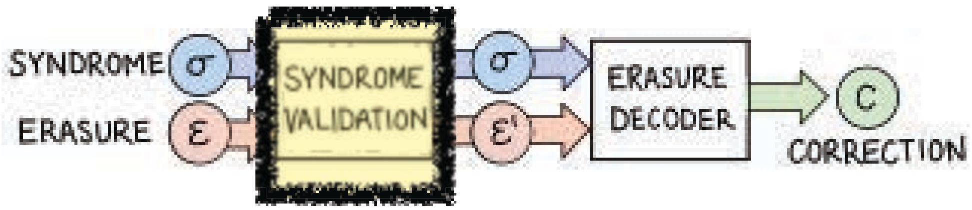
while there exists an odd cluster:

Iterate over all odd clusters:

1. Grow the cluster by a half-edge
2. If the cluster meets another cluster, fuse and update parity
3. If the new cluster is even, remove it from the odd cluster list

Add any edges that are full grown to the list of erased edges.





Syndrome Validation

Syndrome Validation

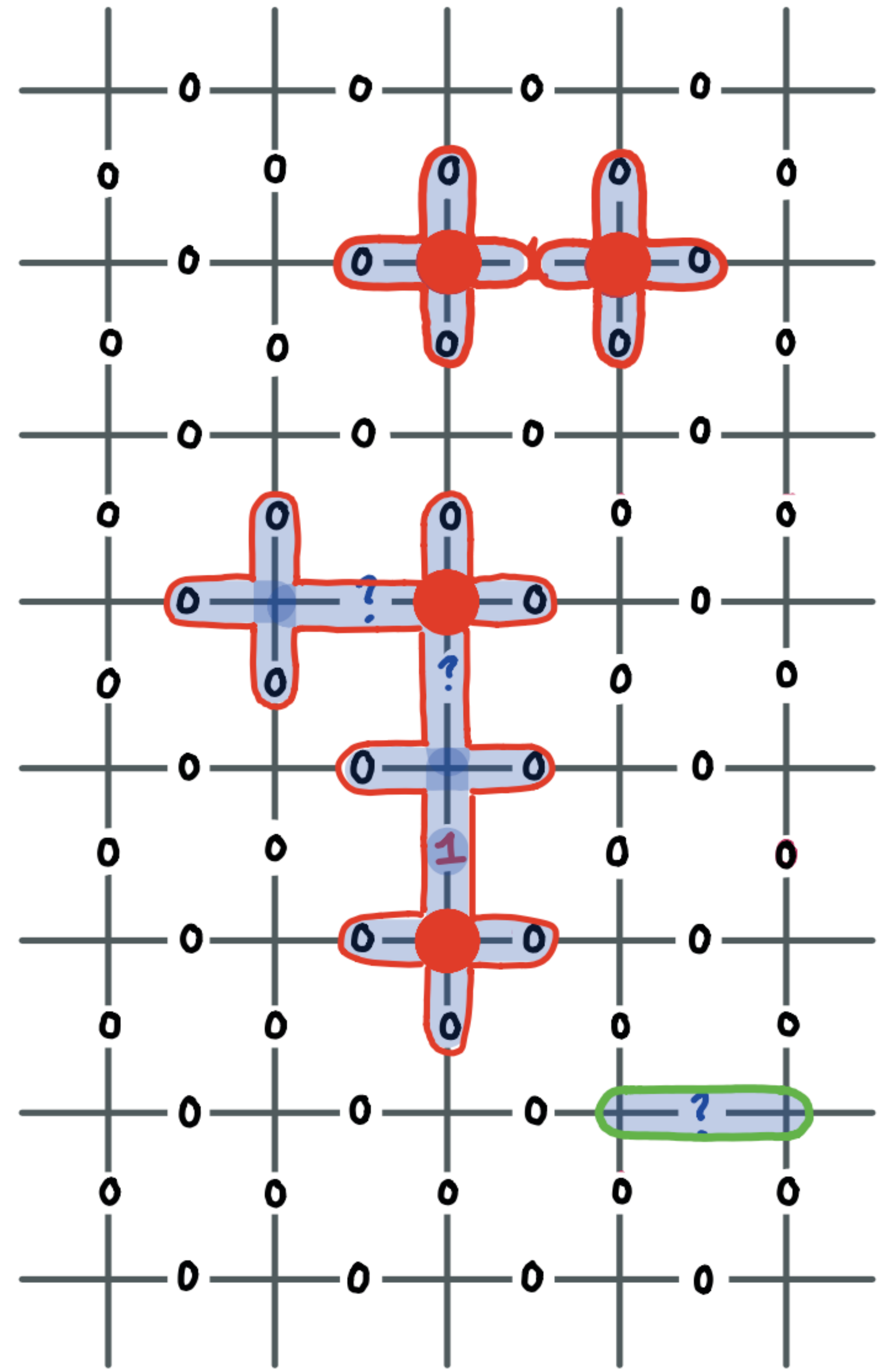
Create a list of all odd clusters

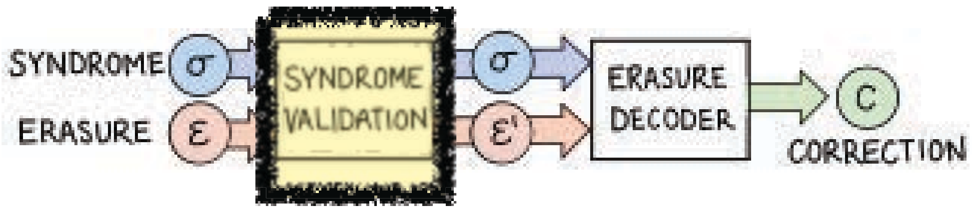
while there exists an odd cluster:

Iterate over all odd clusters:

1. Grow the cluster by a half-edge
2. If the cluster meets another cluster, fuse and update parity
3. If the new cluster is even, remove it from the odd cluster list

Add any edges that are full grown to the list of erased edges.



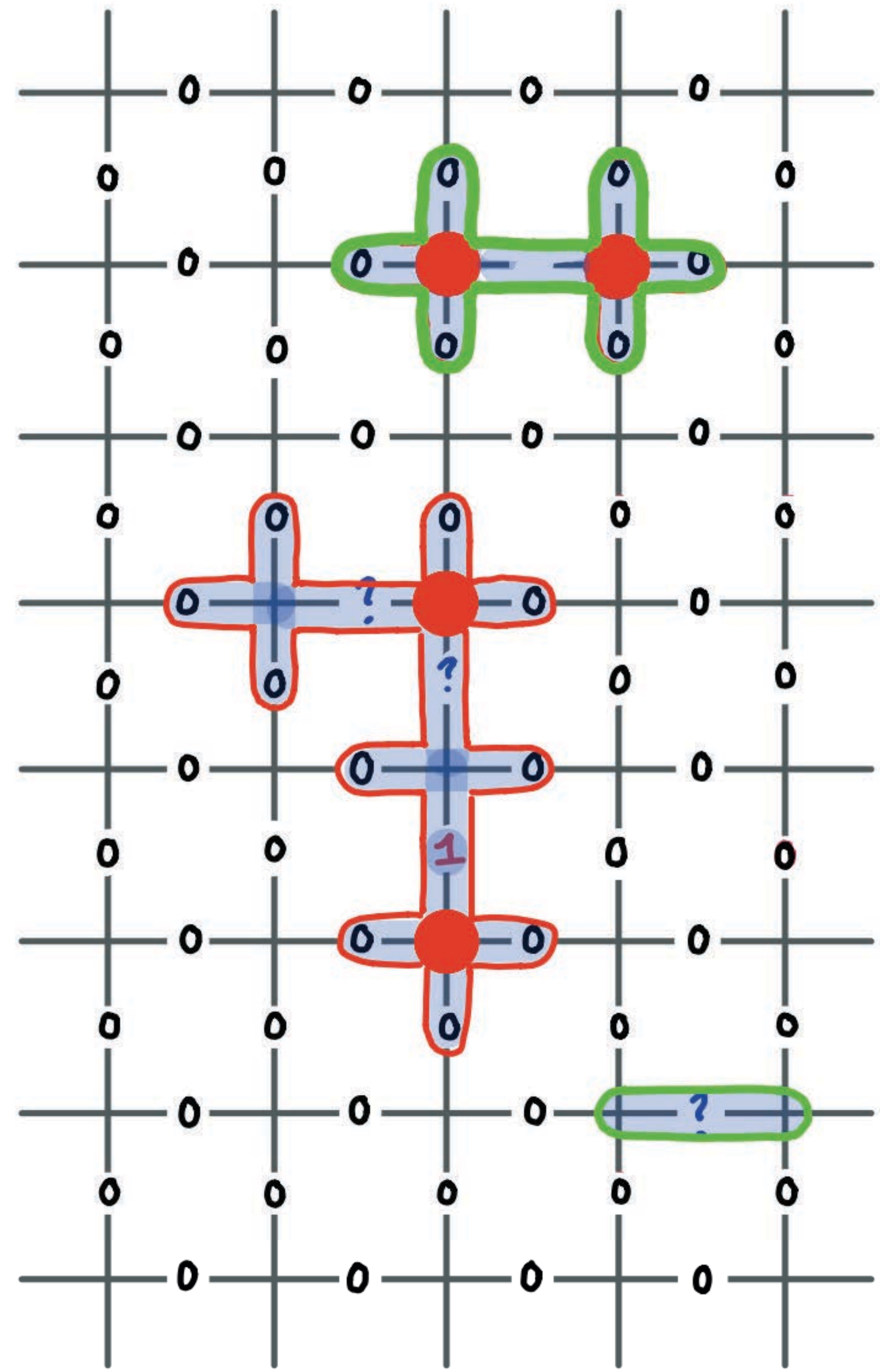


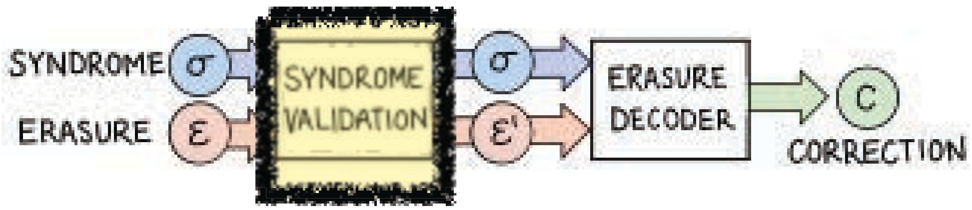
Syndrome Validation

Syndrome Validation

Create a list of all odd clusters

```
while there exists an odd cluster:
    Iterate over all odd clusters:
        1. Grow the cluster by a half-edge
        2. If the cluster meets another cluster, fuse and
           update parity
        3. If the new cluster is even, remove it from the
           odd cluster list
    Add any edges that are full grown to the list of erased
edges.
```





Syndrome Validation

Syndrome Validation

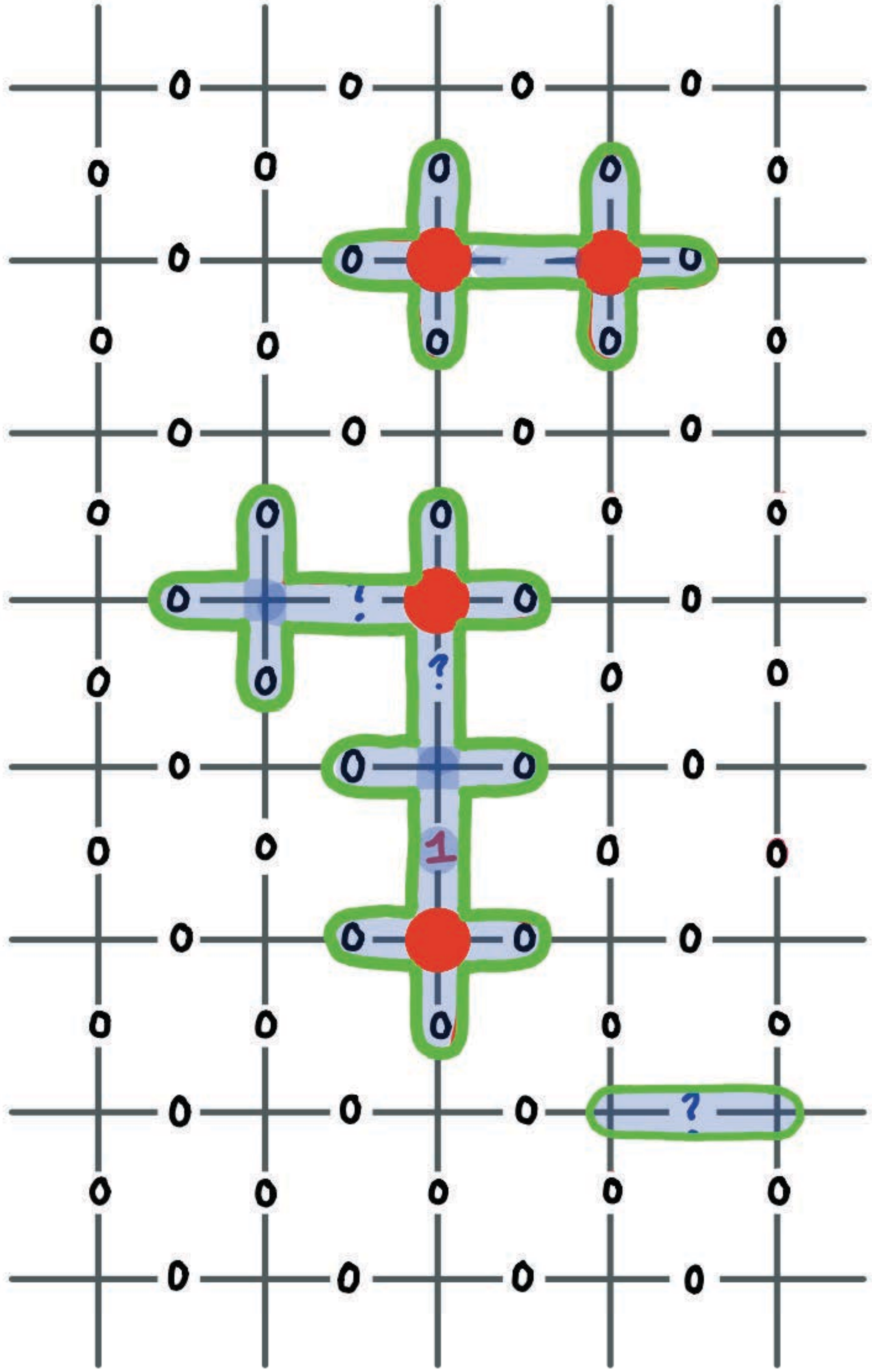
Create a list of all odd clusters

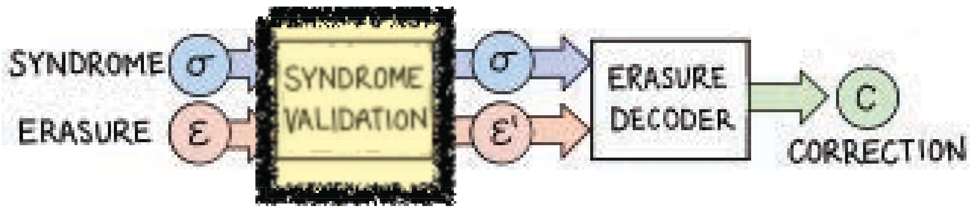
while there exists an odd cluster:

Iterate over all odd clusters:

1. Grow the cluster by a half-edge
2. If the cluster meets another cluster, fuse and update parity
3. If the new cluster is even, remove it from the odd cluster list

Add any edges that are full grown to the list of erased edges.





Syndrome Validation

Syndrome Validation

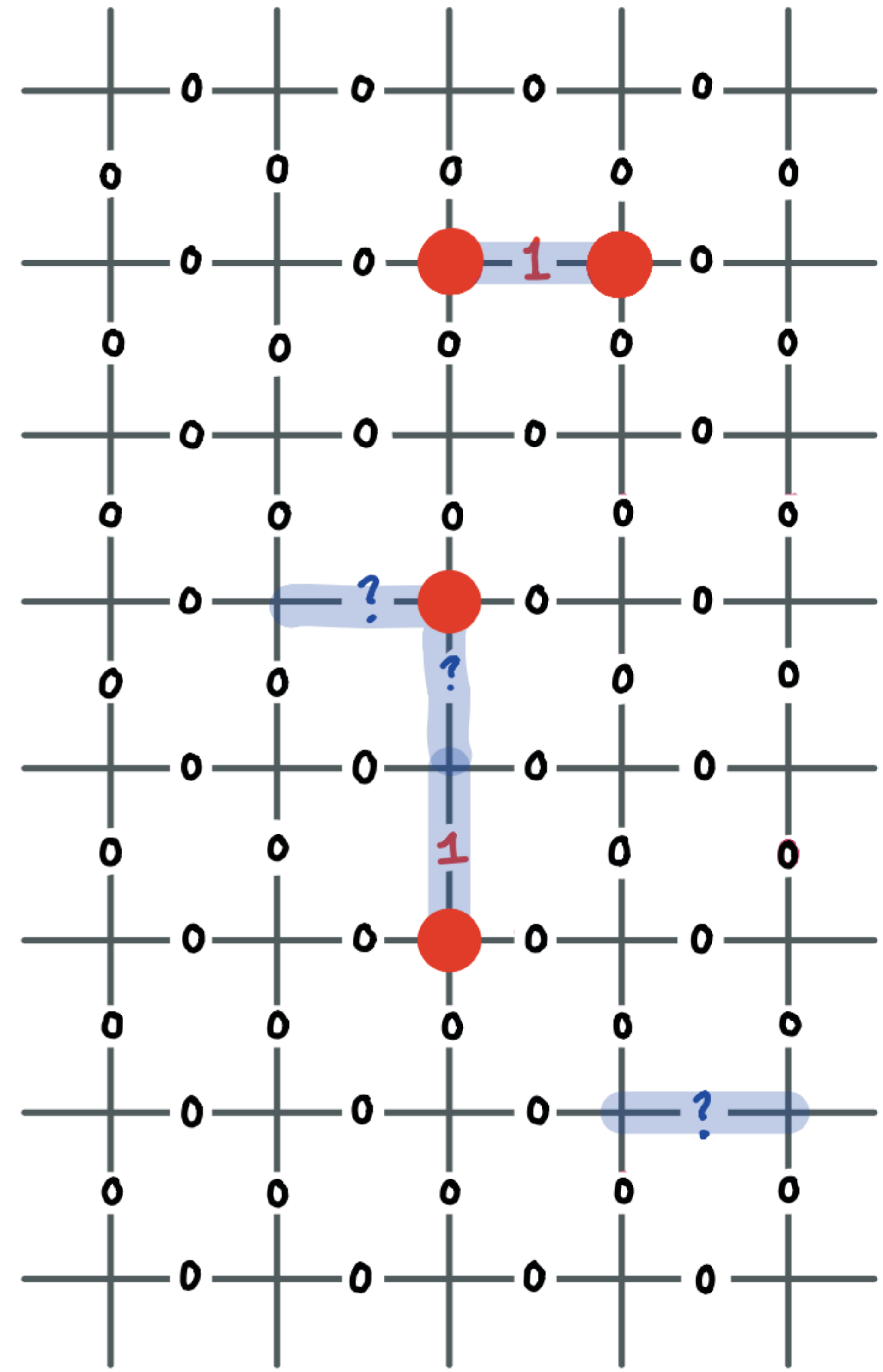
Create a list of all odd clusters

while there exists an odd cluster:

    Iterate over all odd clusters:

1. Grow the cluster by a half-edge
2. If the cluster meets another cluster, fuse and update parity
3. If the new cluster is even, remove it from the odd cluster list

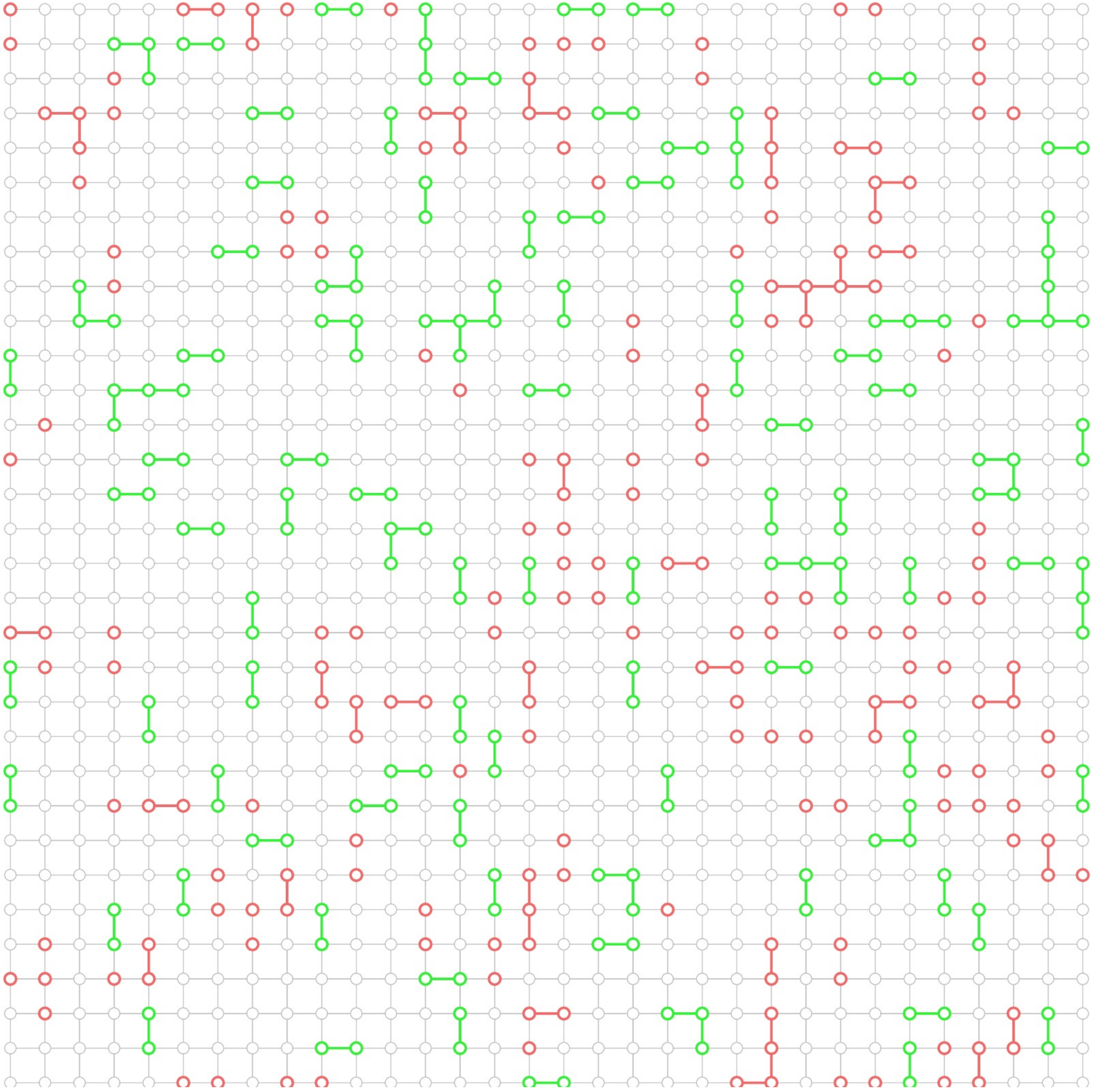
    Add any edges that are full grown to the list of erased edges.



Example: decoding with 10% erasure, 5% Pauli error

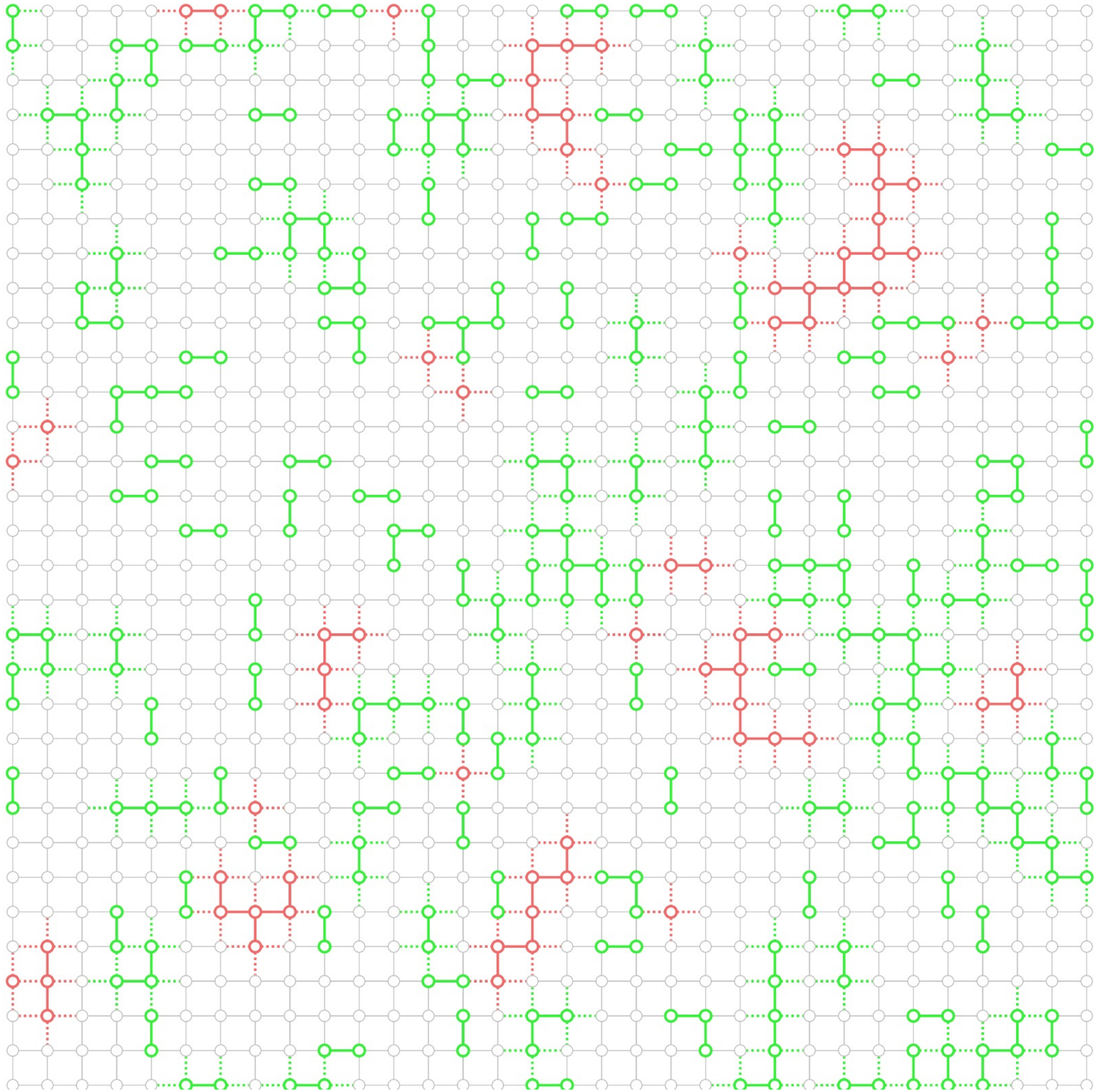
Example: decoding with 10% erasure, 5% Pauli error

Measure Syndromes



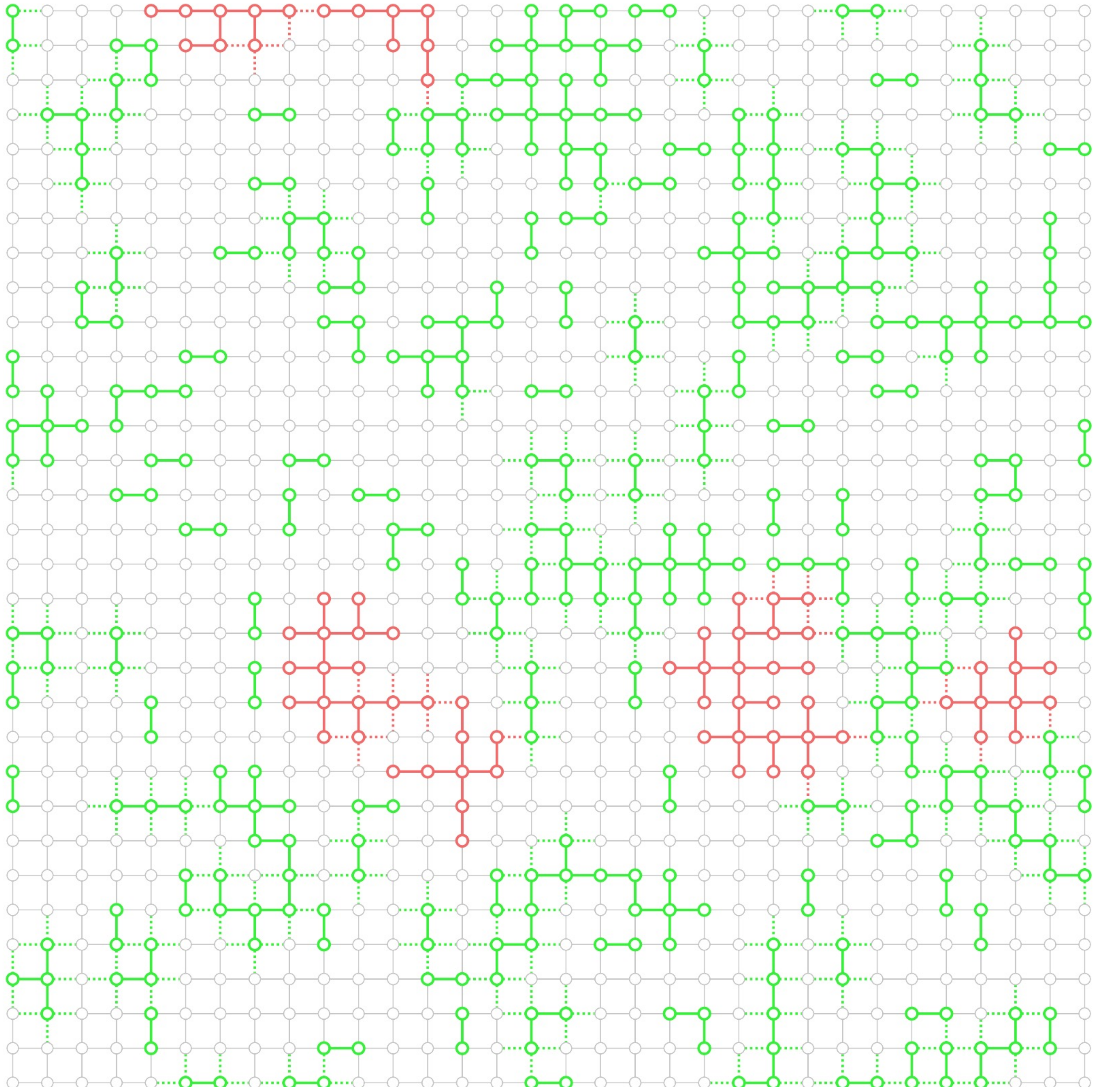
Example: decoding with 10% erasure, 5% Pauli error

Syndrome Validation

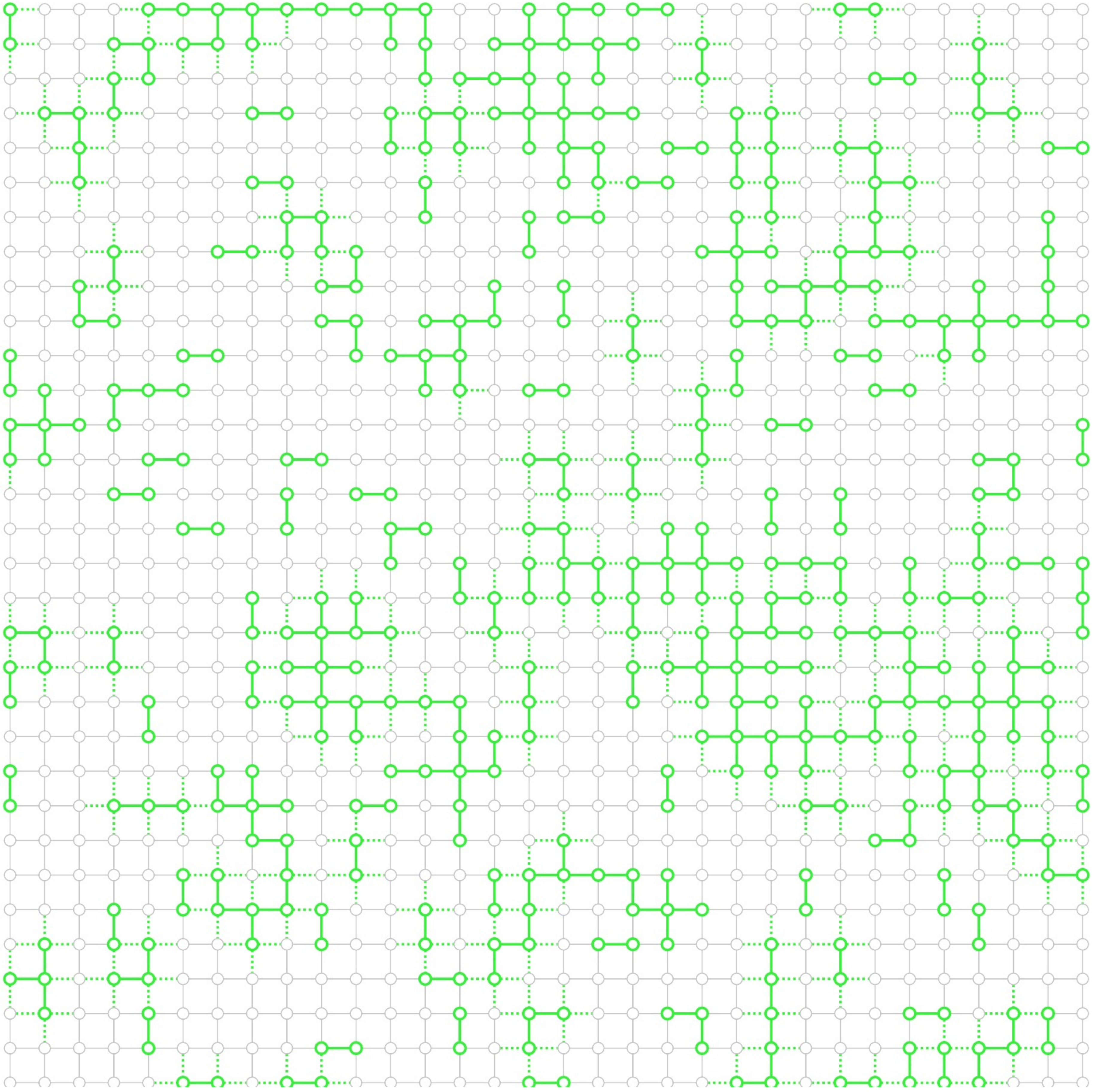


Example: decoding with 10% erasure, 5% Pauli error

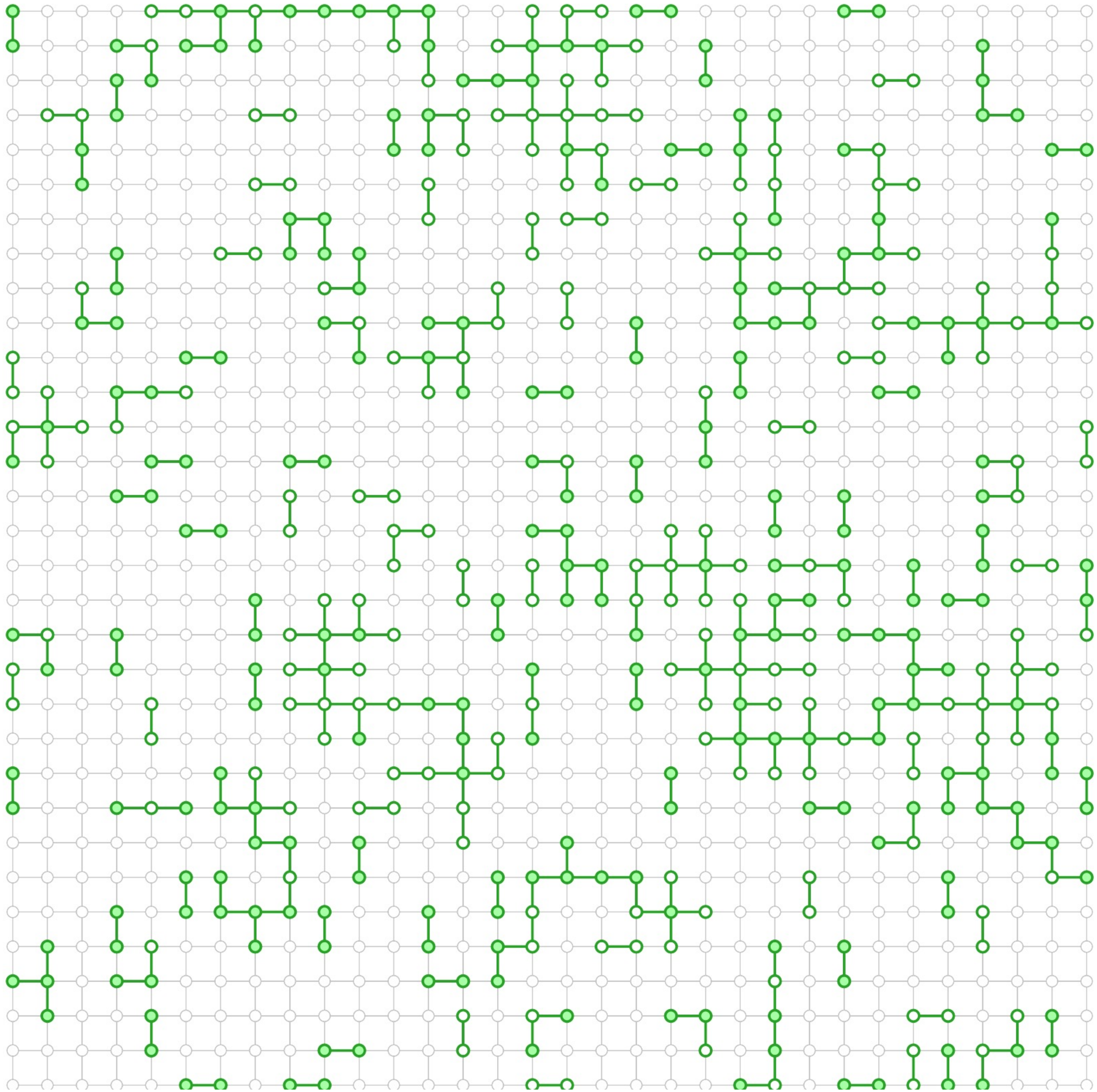
Syndrome Validation



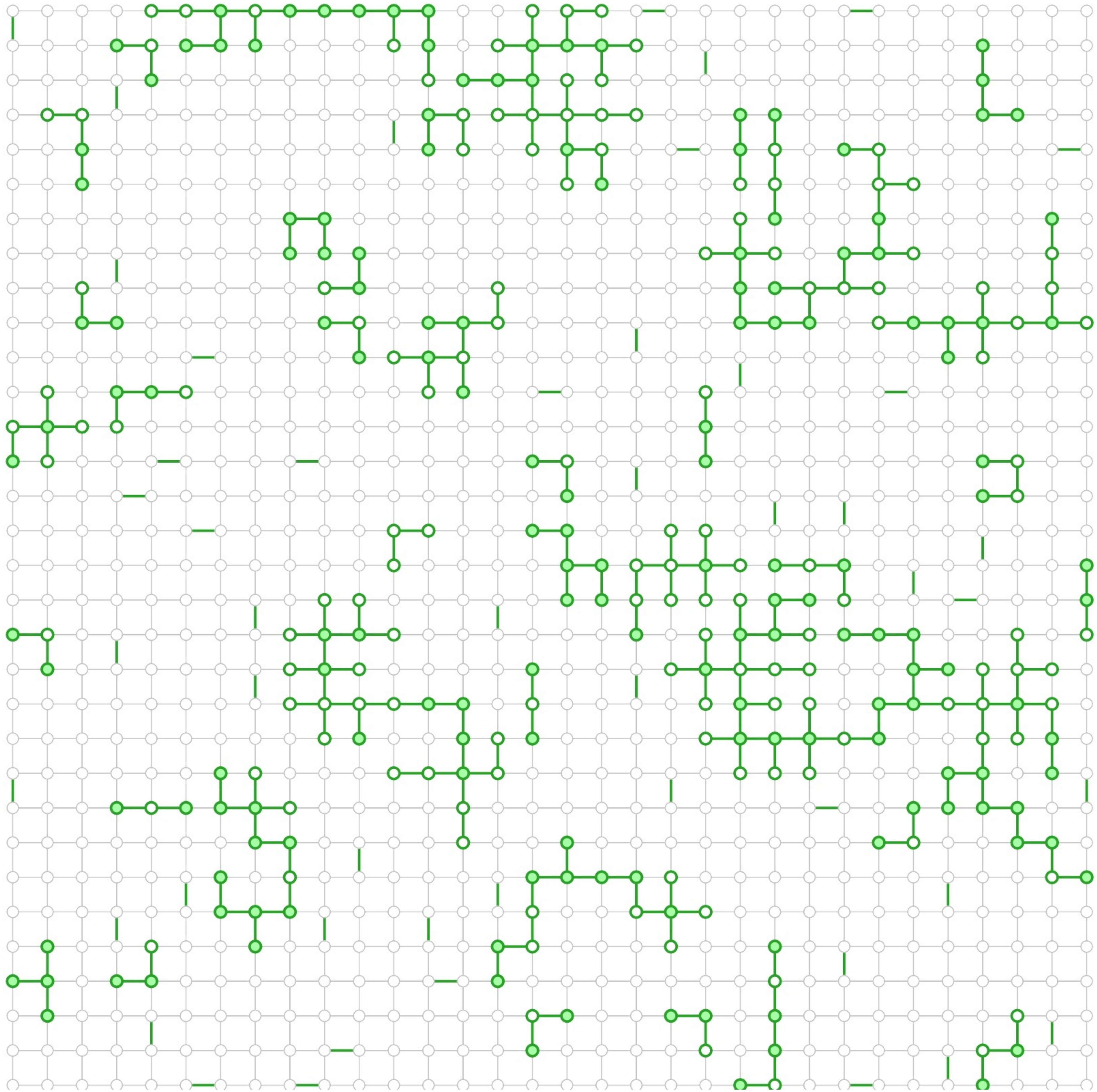
Example: decoding with 10% erasure, 5% Pauli error



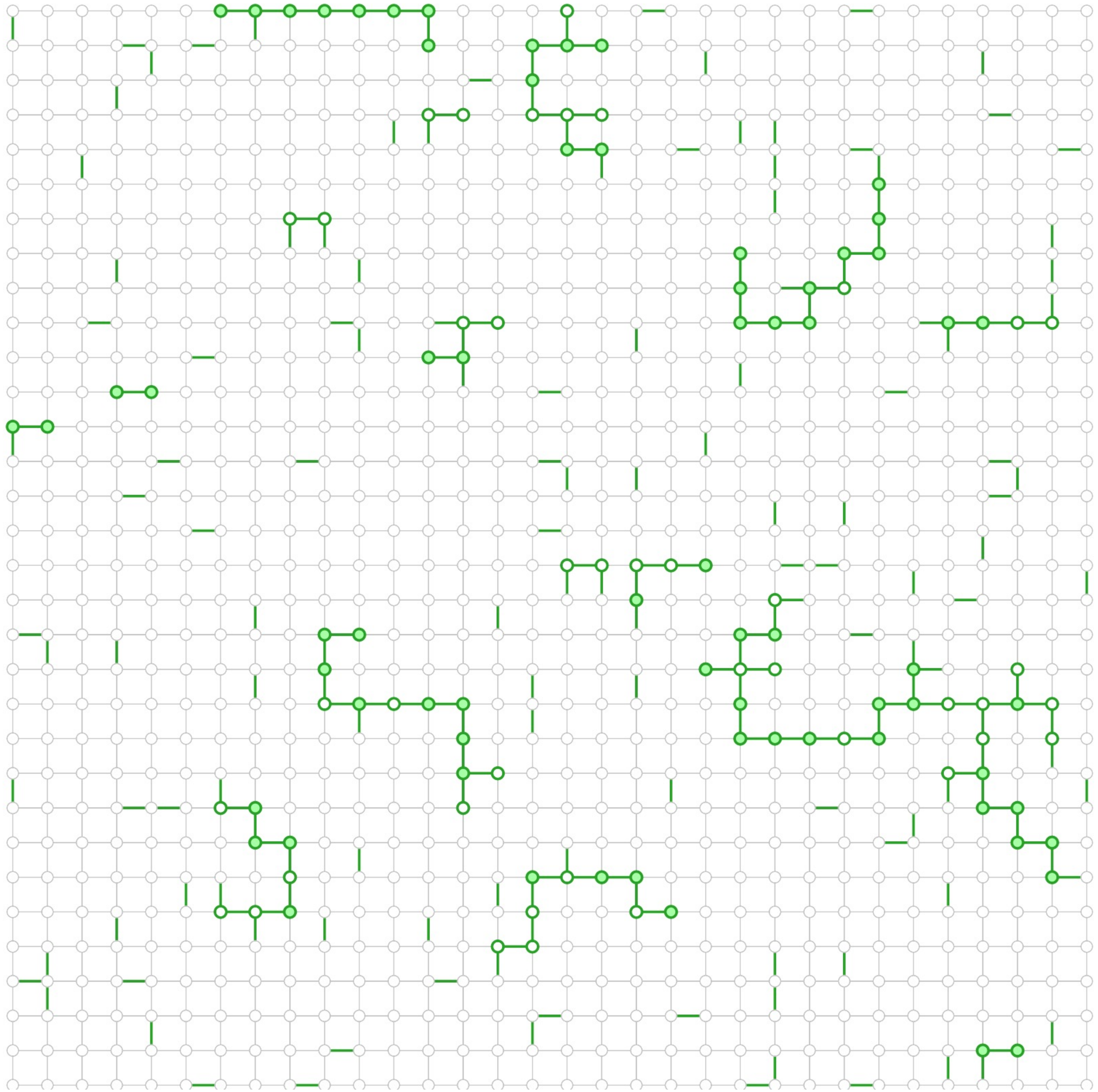
Example: decoding with 10% erasure, 5% Pauli error



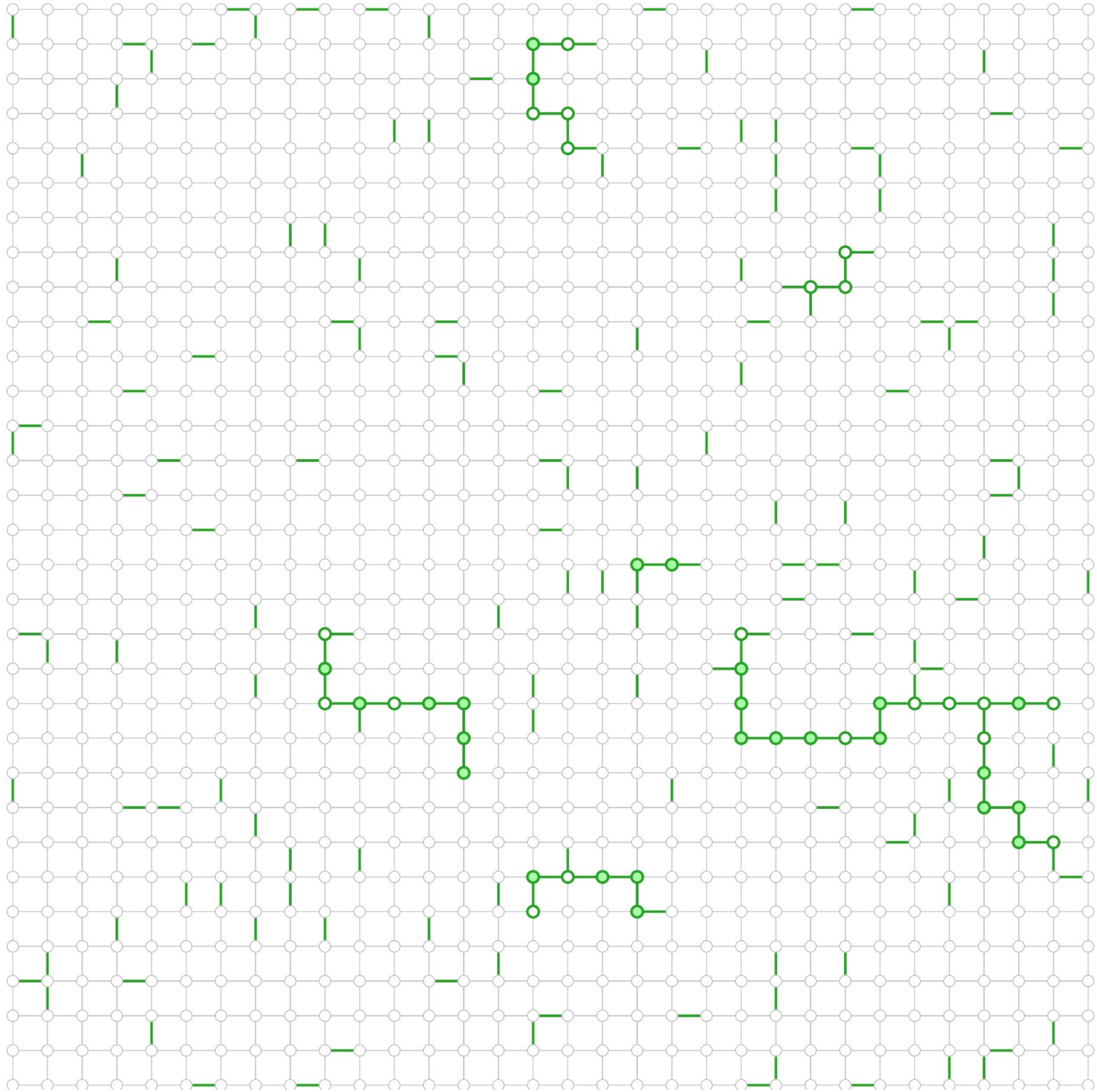
Example: decoding with 10% erasure, 5% Pauli error



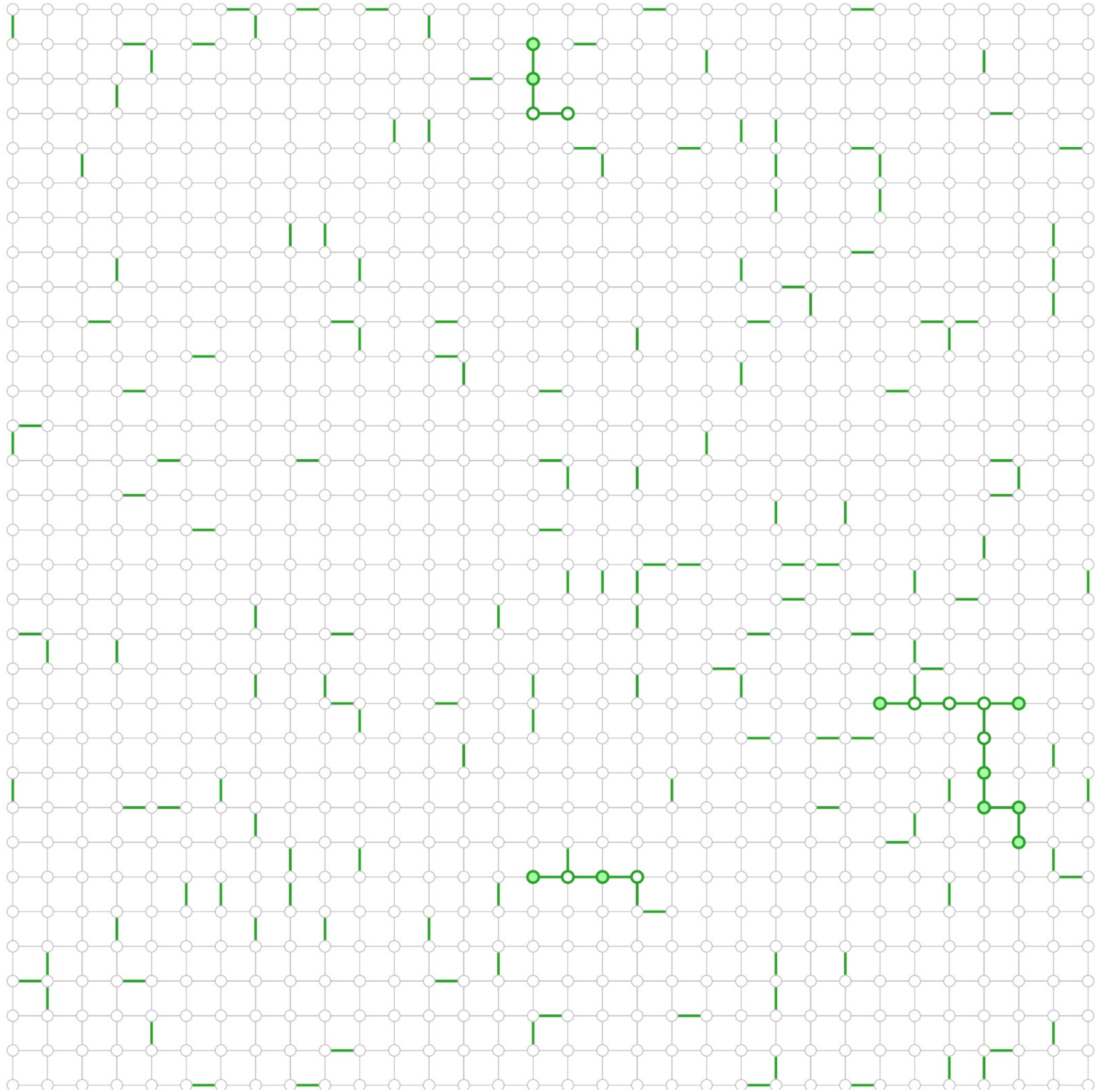
Example: decoding with 10% erasure, 5% Pauli error



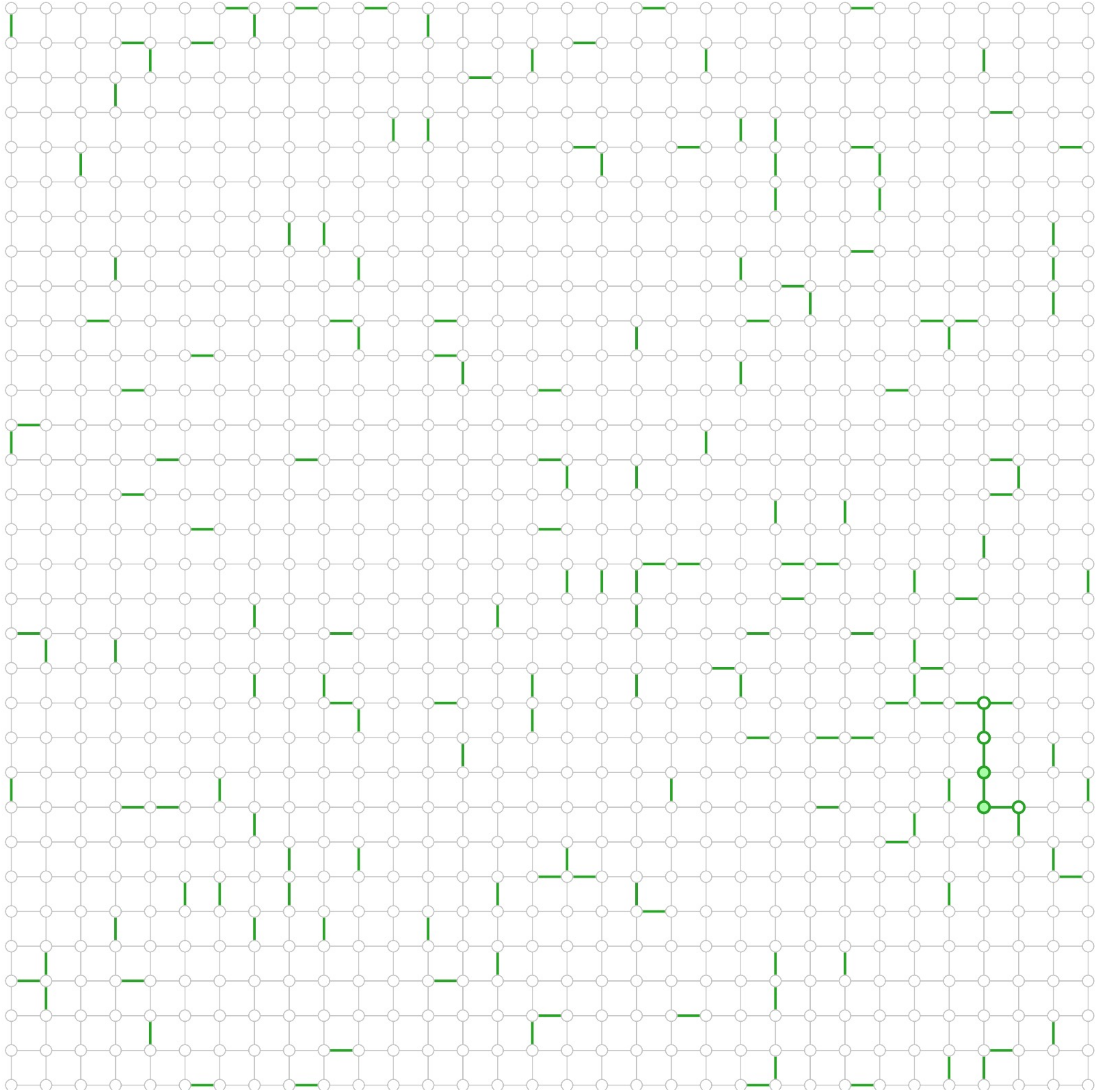
Example: decoding with 10% erasure, 5% Pauli error



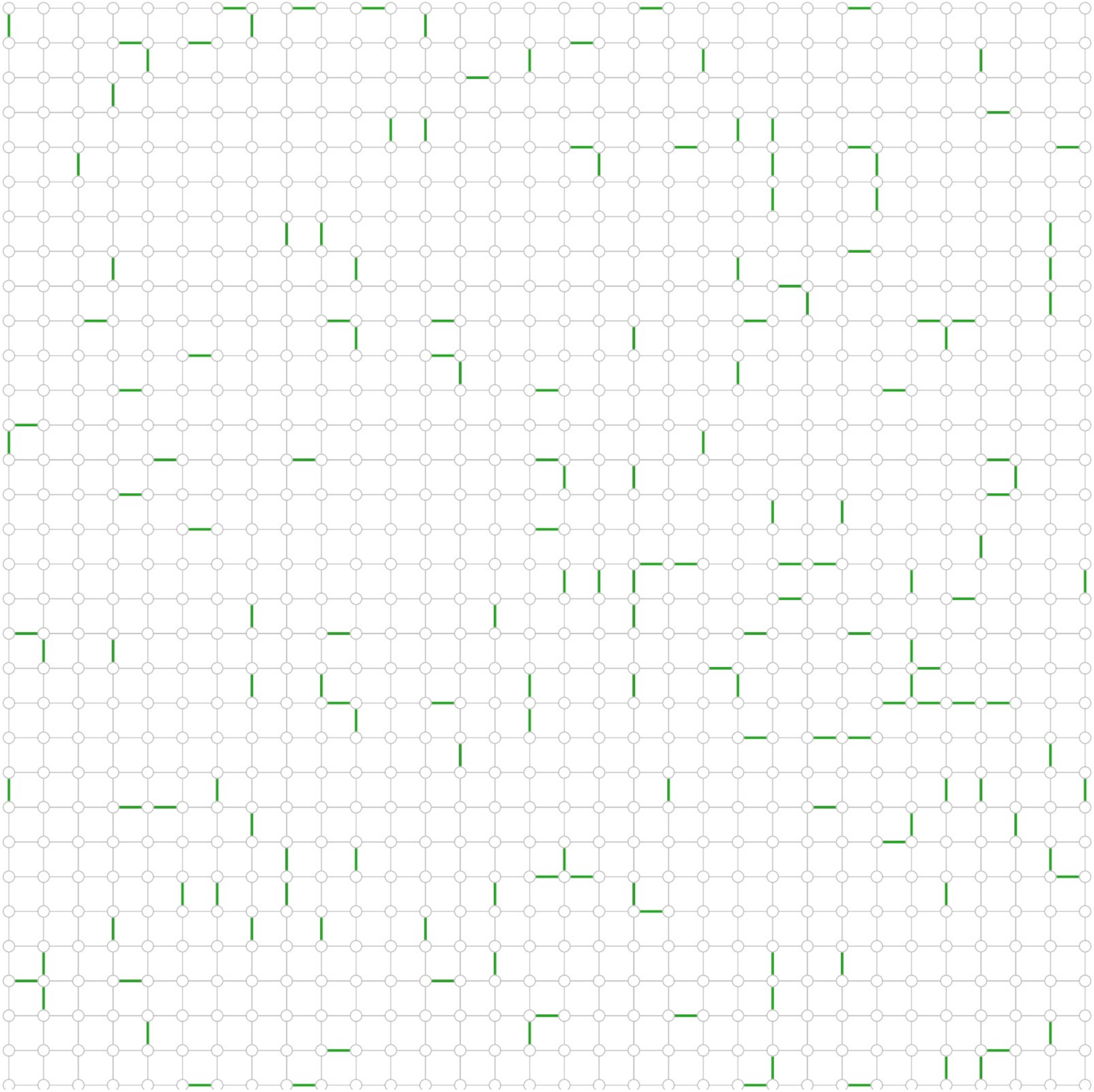
Example: decoding with 10% erasure, 5% Pauli error



Example: decoding with 10% erasure, 5% Pauli error

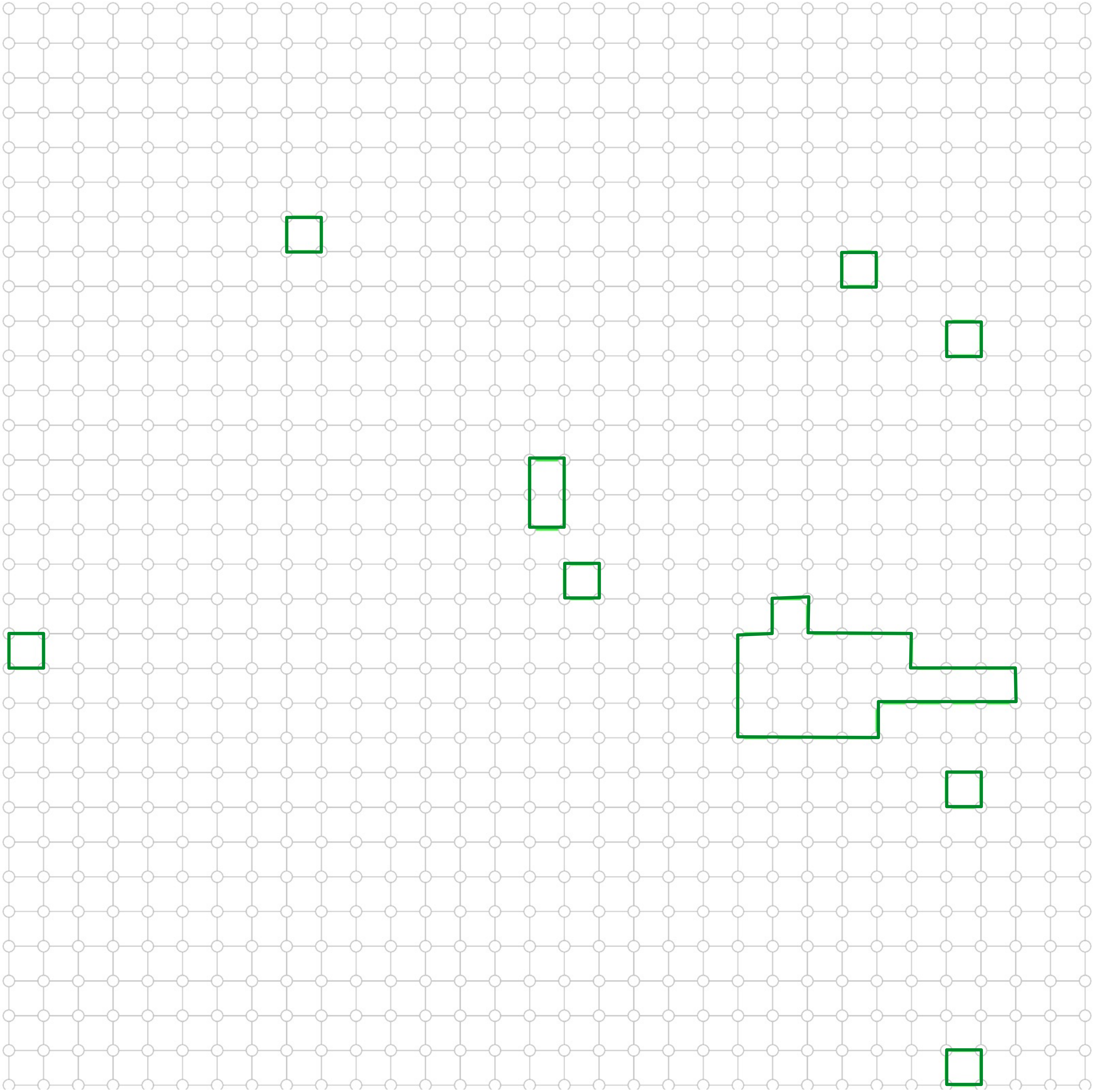


Example: decoding with 10% erasure, 5% Pauli error



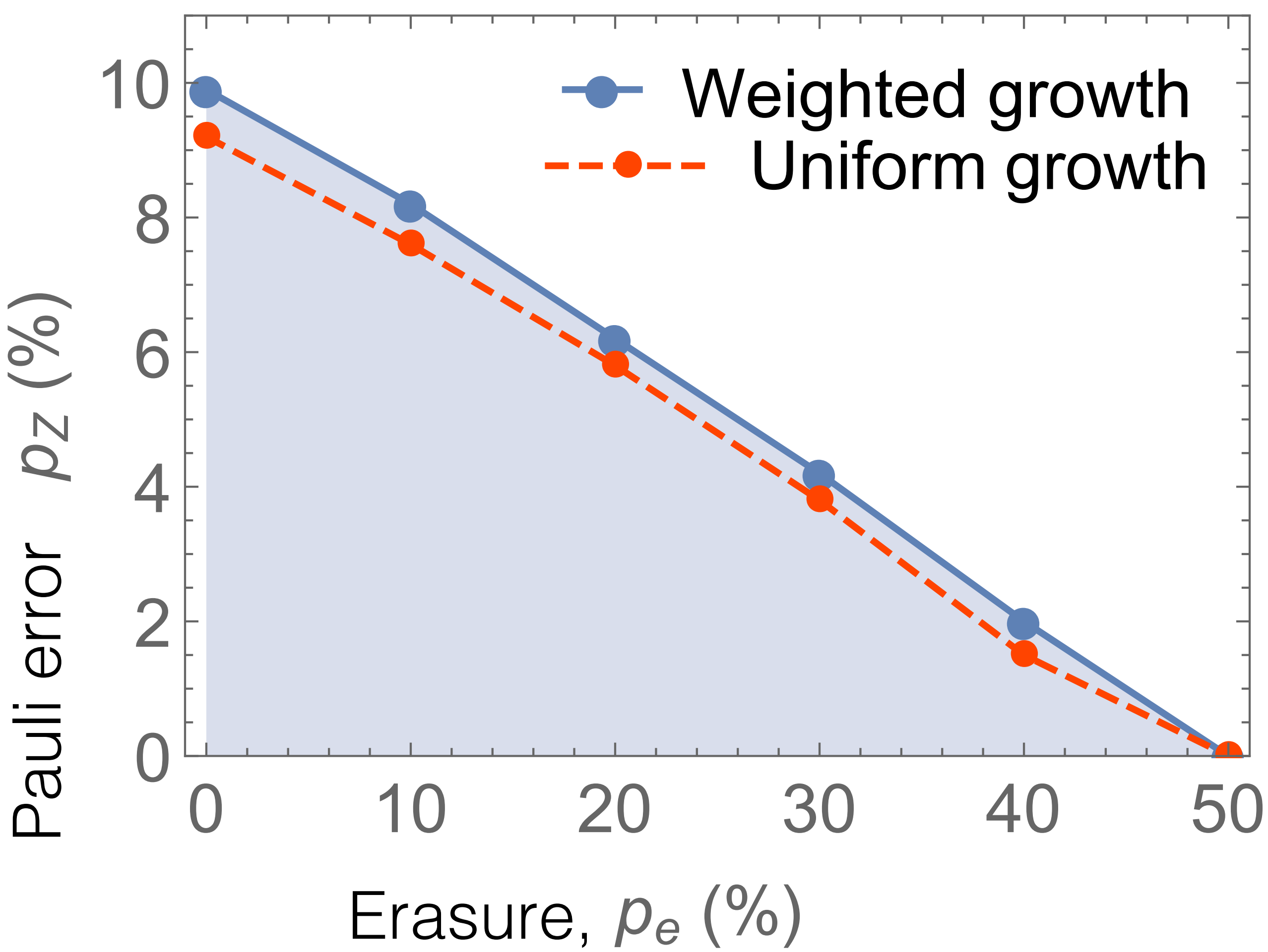
Example: decoding with 10% erasure, 5% Pauli error

Resulting error



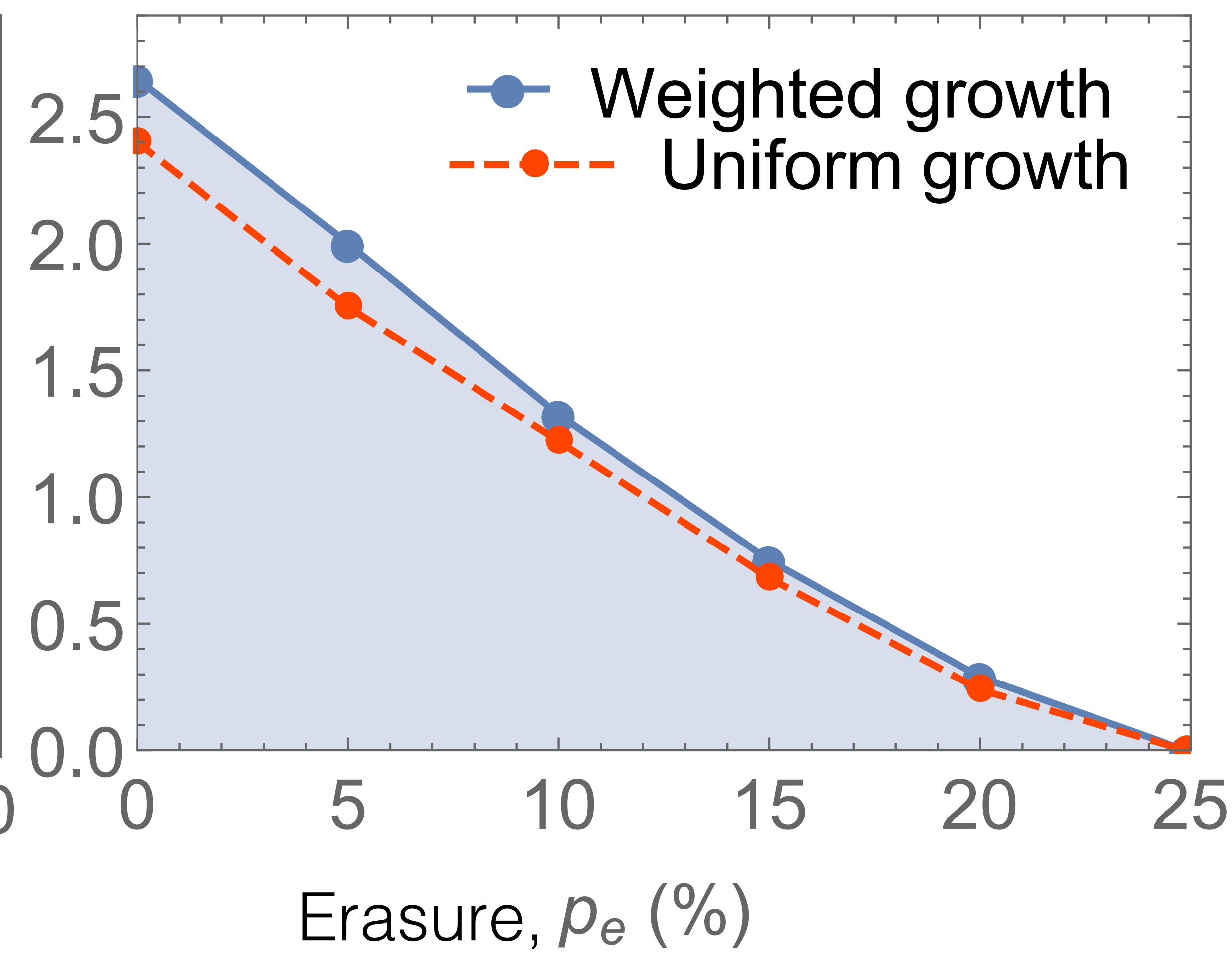
Performance: Threshold

2D surface code with bit-flip errors



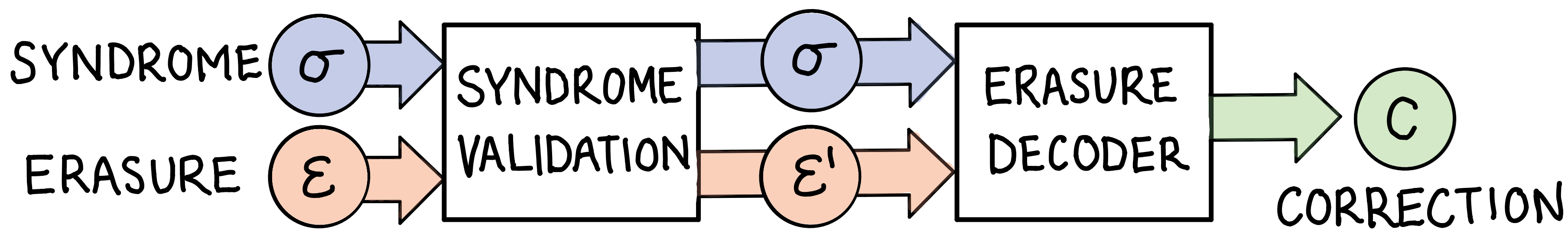
Pauli error threshold:
9.9%

2+1D surface code, phenomenological noise



Phenomenological error threshold:
2.6%

Wider Applicability



- **Fault tolerance:** Exactly the same algorithm will decode the 2+1d surface code, which has a cubic syndrome graph.

- **Arbitrary surface code:** Method works for any structure of syndrome graph, and can be applied directly to any surface code, including those with unusual geometry such as hyperbolic codes.

- **Color code:** By projecting the surface code onto the color code, an arbitrary color code can be decoded.

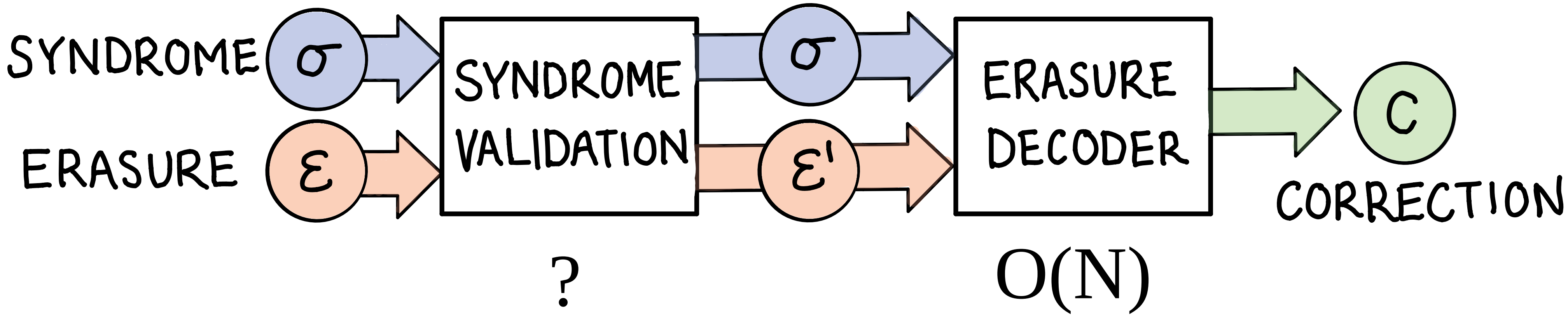
- **Other codes?**: For any code for which there is an erasure decoder, and a notion of distance between syndromes, this approach can be used to create a decoder for Pauli error.

1. What is decoding and how fast does it need to be?

2. The Union Find decoder

3. Achieving almost-linear time

Achieving almost linear complexity

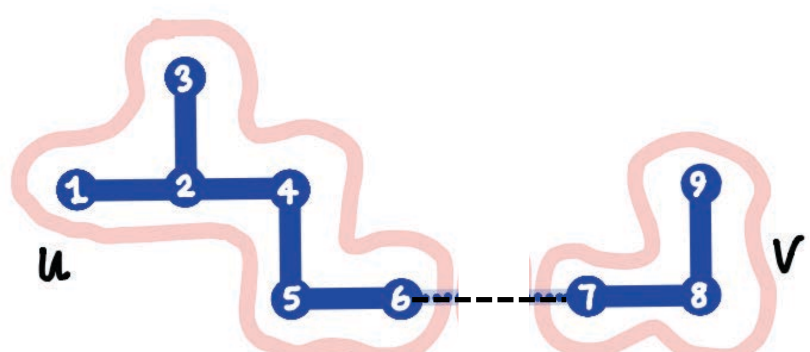


UNION(u,v)

Merges clusters u and v

FIND(n)

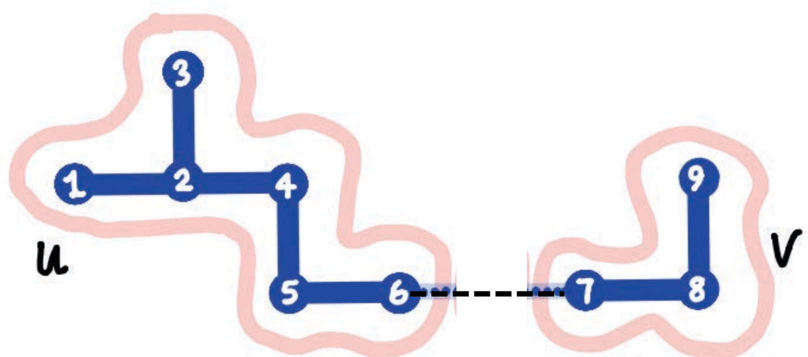
Returns the cluster to which node n belongs



Achieving almost linear complexity

Naive algorithm

Data structure: Lookup table for each node



node:	1	2	3	4	5	6	7	8	9
cluster	u	u	u	u	u	u	v	v	v

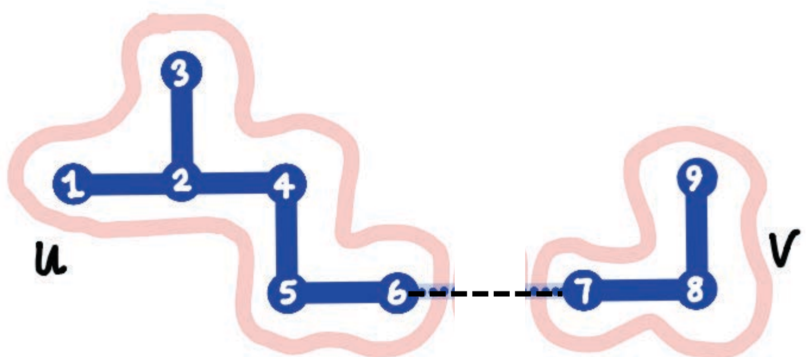
FIND: Lookup node: O(1)
UNION: Relabel every element of one cluster: O(N)

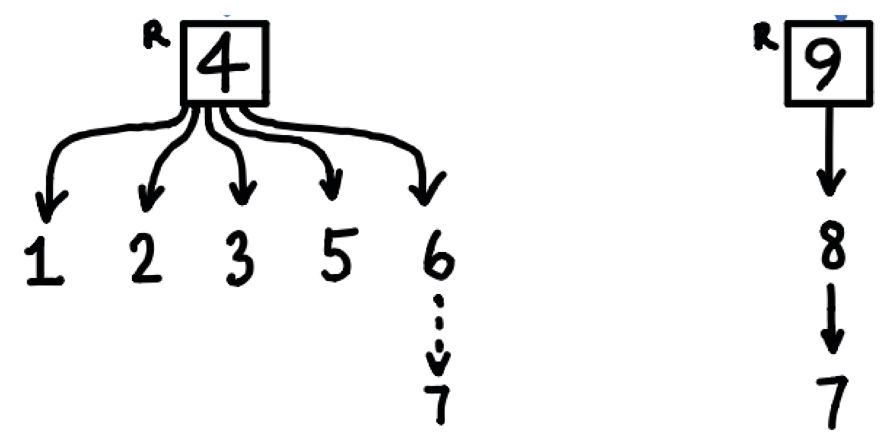
Worst case complexity: $O(N^2)$

Achieving almost linear complexity

Better algorithm

Data structure: Tree, stored as a linked list
root of tree identifies the cluster





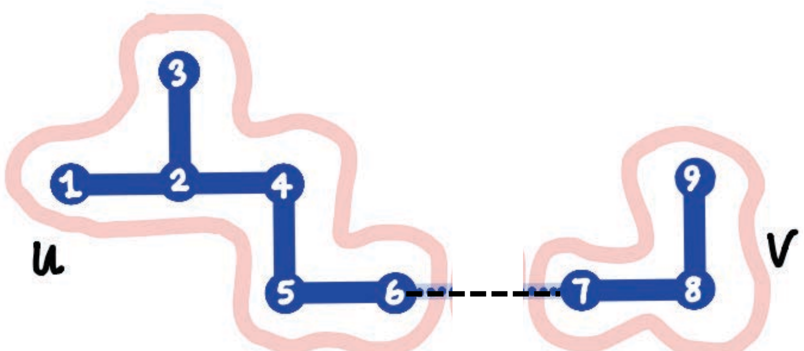
FIND: Traverse tree to find root: O(log N)
UNION: Point root of one cluster to the other O(1)

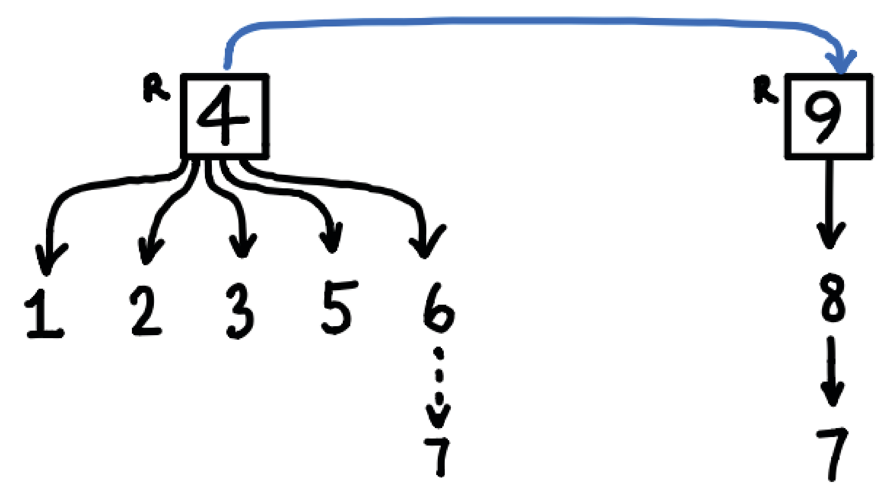
Worst case complexity: O(N log N)

Achieving almost linear complexity

Better algorithm

Data structure: Tree, stored as a linked list
root of tree identifies the cluster





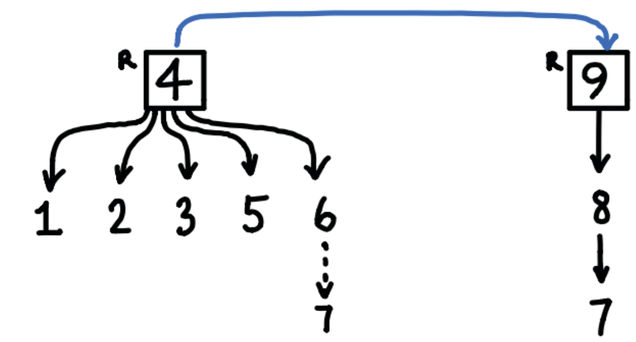
FIND: Traverse tree to find root: O(log N)
UNION: Point root of one cluster to the other O(1)

Worst case complexity: O(N log N)

Achieving almost linear complexity

Even better algorithm

Data structure: Tree, stored as a linked list
root of tree identifies the cluster



+ Weighted Union

During UNION always updates the smallest of the two clusters.
Size of smaller cluster at least doubles when UNION is called.

+ Path compression

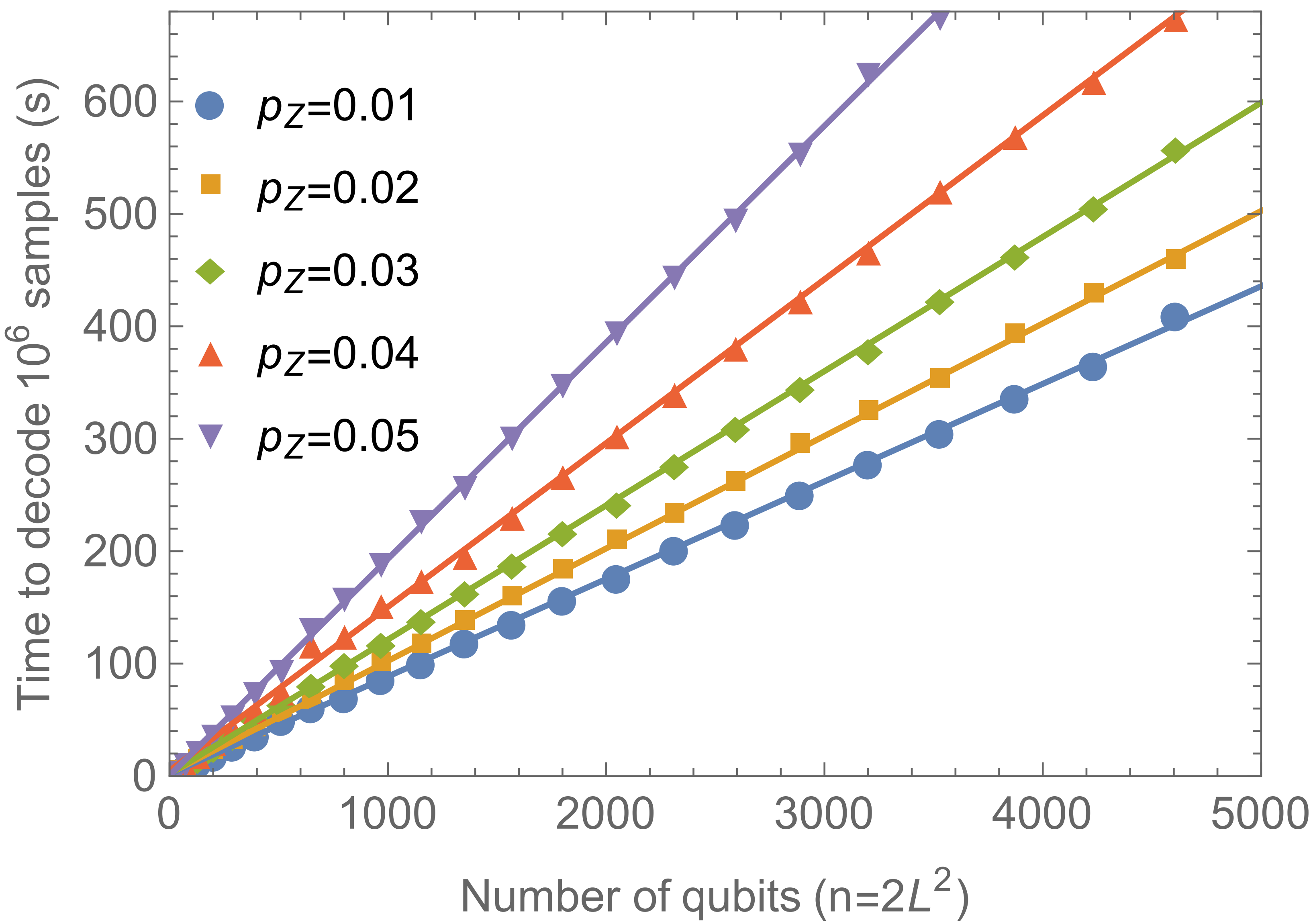
After FIND(u) Is called, add a new edge pointing u directly to the root.
If Find(u) is called again, it will take 1 step to return the root.

The analysis of these three things combined was first made by Tarjan:
R. E. Tarjan, Journal of the ACM (JACM) 22, 215. (1975).

Worst case complexity: $O(\alpha(N)\, N)$

$\alpha(N)$ is the inverse of Ackermann's function, and $\alpha(N) \leq 3$ as long as N < $2^{2^{2^{\cdot^{\cdot^{2}}}}}$ with 65536 twos.

Performance: Running time



Why you should care about the Union Find Decoder if:

You want to build a quantum computer:

- The UF decoder is very fast in practice, with effectively linear scaling and a small constant overhead
- Very simple algorithm, good for implementing in hardware.

You want to numerically study codes with unusual geometries:

- The decoder can be applied to any surface code (2d or 2+1d), without any adaptation, to color codes, and potentially to wider classes of codes.

You want to understand the connection between erasure errors and Pauli errors:

- The decoding algorithm provides one way of converting Pauli errors into erasures. Maybe this can help us better understand how they are related?

Questions still to answer:

- How fast can it run in hardware?
- Compare directly to other decoders’ below threshold performance
- What’s the best way to parallelize the algorithm?
- Can the threshold be further improved?
- Can the algorithm be adapted to account for more types of errors?