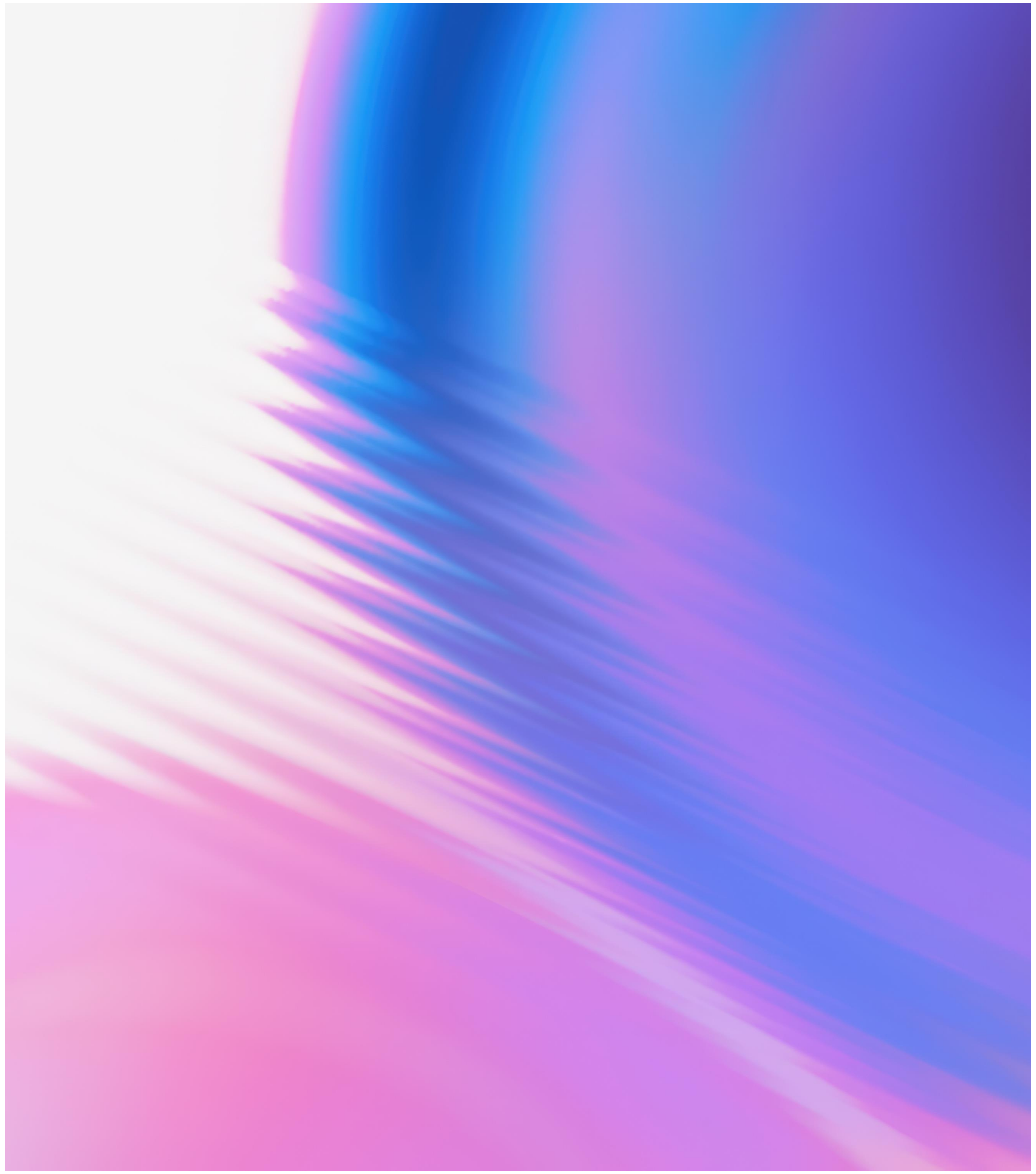
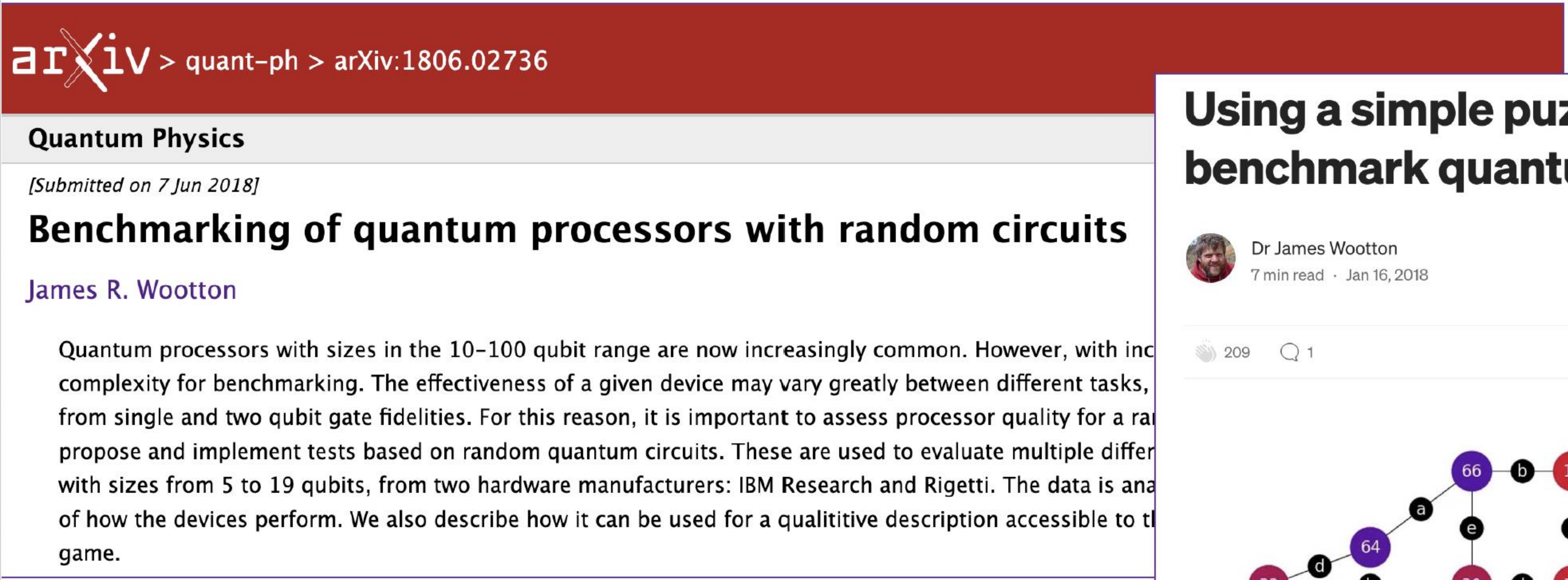
Quantum puzzles on utility scale devices

James R. Wootton

Moth Quantum (now)
IBM Quantum (2018-2024)
University of Basel (2016-2018)



Resurrecting work from 6 years ago



arXiv > quant-ph > arXiv:1806.02736

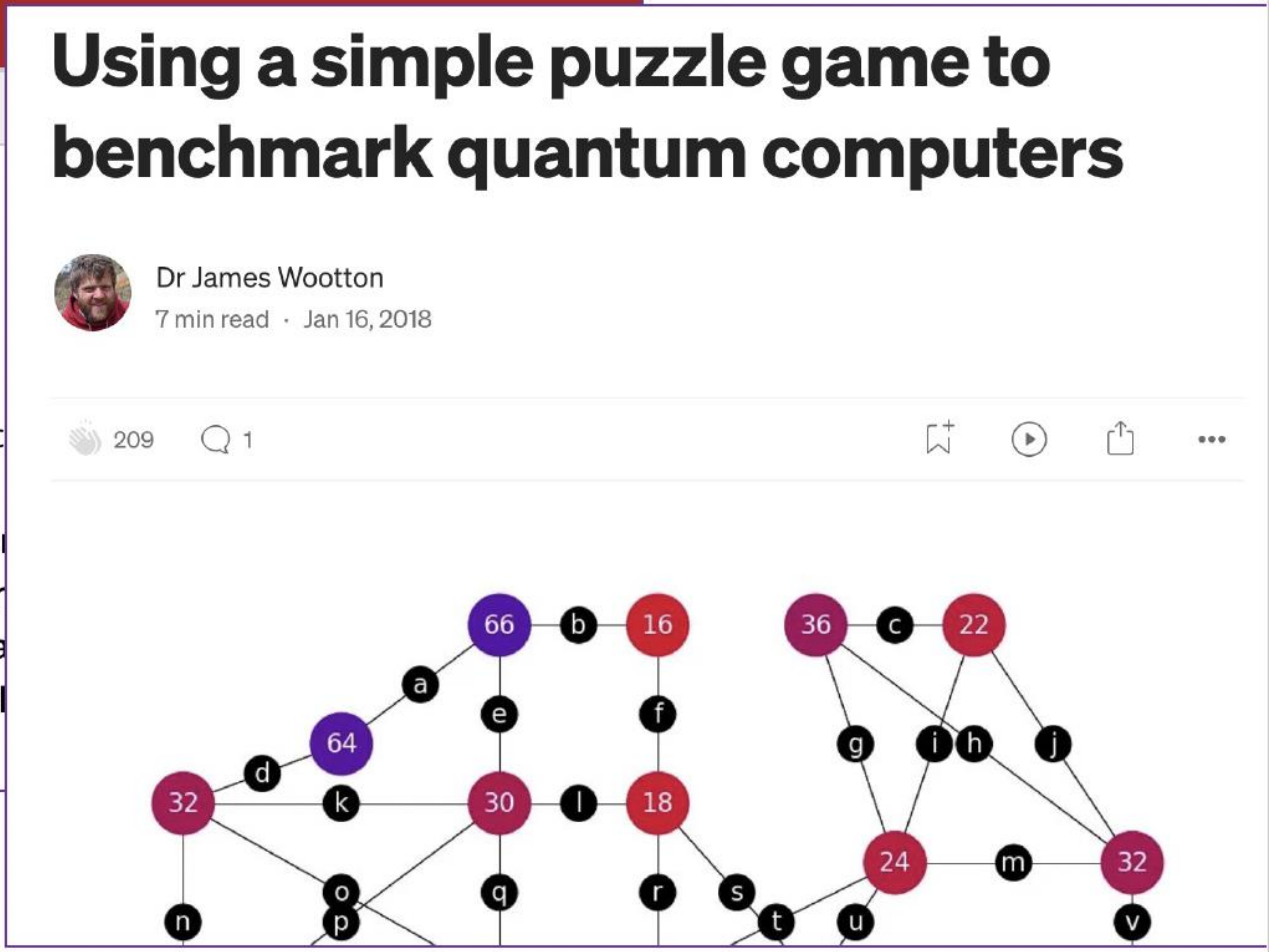
Quantum Physics

[Submitted on 7 Jun 2018]

Benchmarking of quantum processors with random circuits

James R. Wootton

Quantum processors with sizes in the 10-100 qubit range are now increasingly common. However, with inc
complexity for benchmarking. The effectiveness of a given device may vary greatly between different tasks,
from single and two qubit gate fidelities. For this reason, it is important to assess processor quality for a ra
propose and implement tests based on random quantum circuits. These are used to evaluate multiple diffe
with sizes from 5 to 19 qubits, from two hardware manufacturers: IBM Research and Rigetti. The data is ana
of how the devices perform. We also describe how it can be used for a qualititive description accessible to t
game.

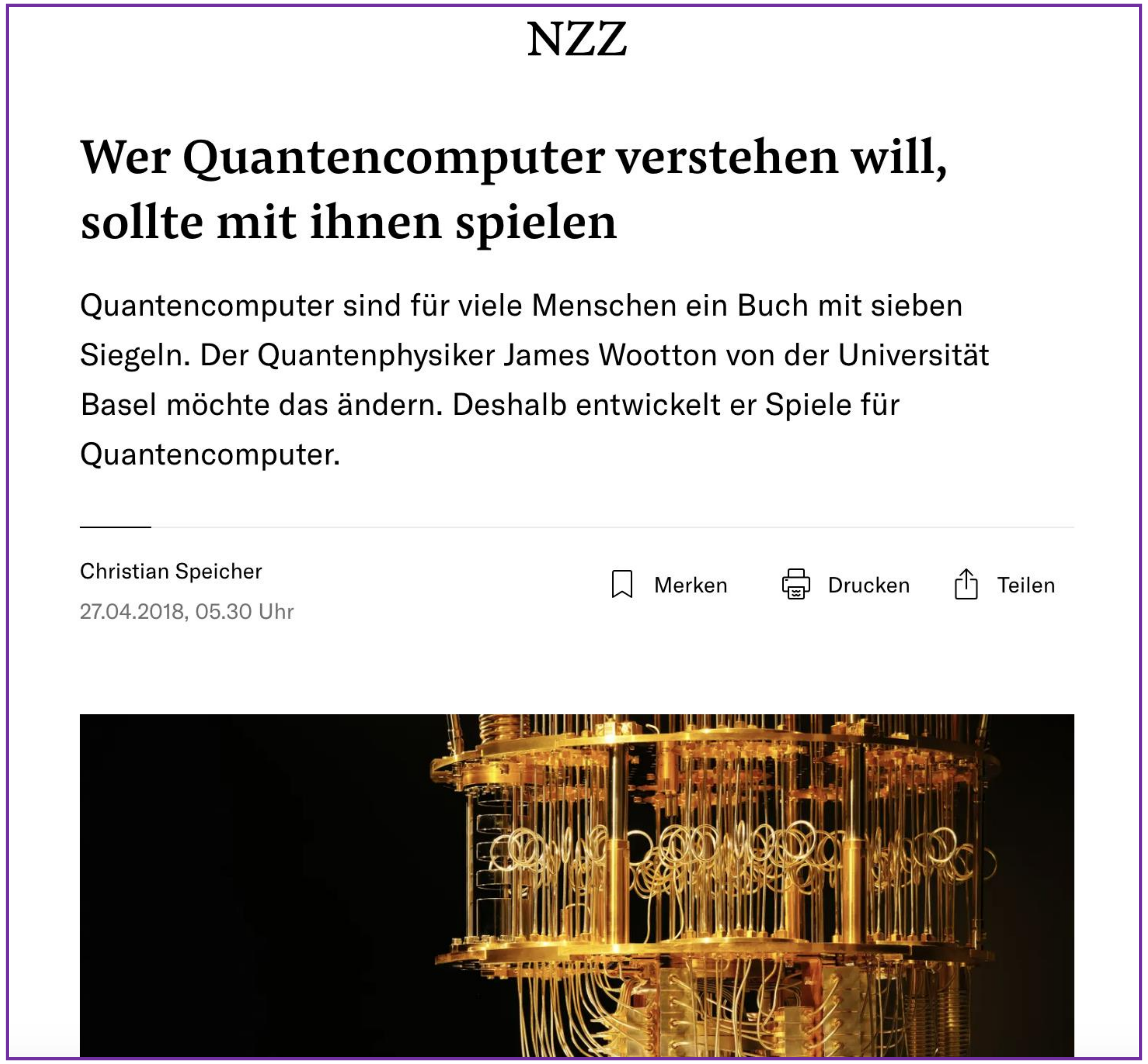


Using a simple puzzle game to benchmark quantum computers

Dr James Wootton
7 min read · Jan 16, 2018

209 1

Resurrecting work from 6 years ago



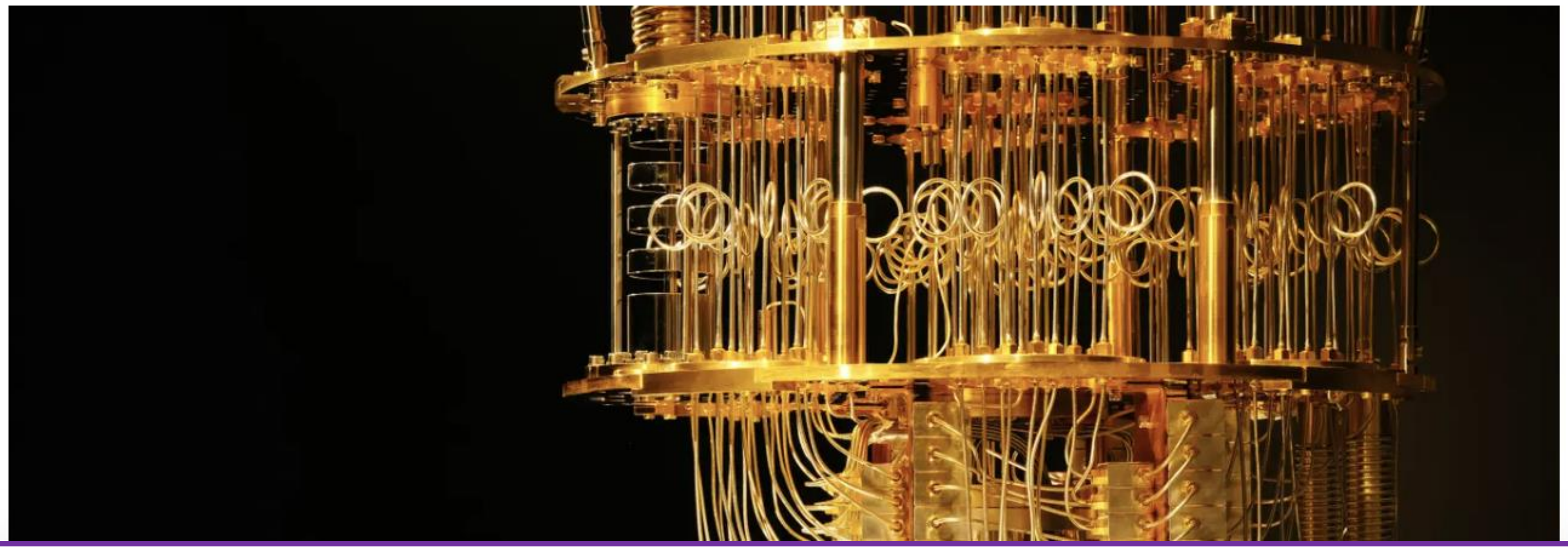
NZZ

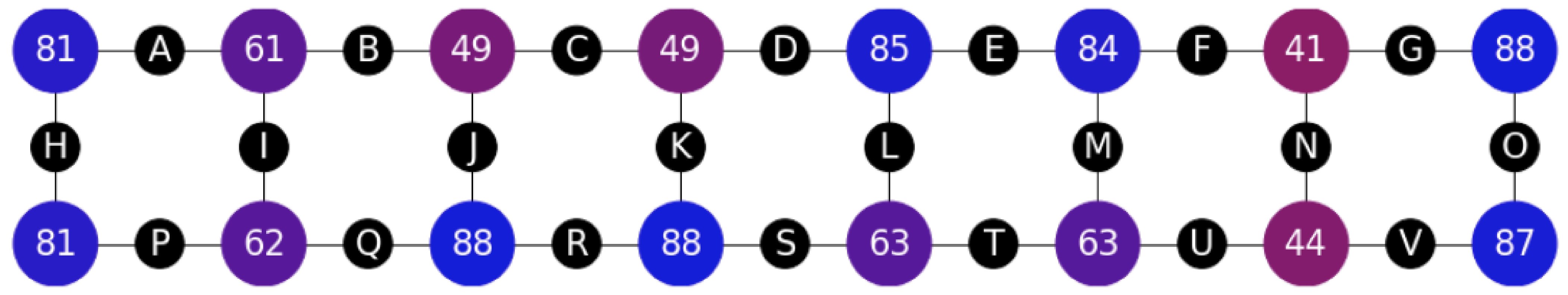
Wer Quantencomputer verstehen will, sollte mit ihnen spielen

Quantencomputer sind für viele Menschen ein Buch mit sieben Siegeln. Der Quantenphysiker James Wootton von der Universität Basel möchte das ändern. Deshalb entwickelt er Spiele für Quantencomputer.

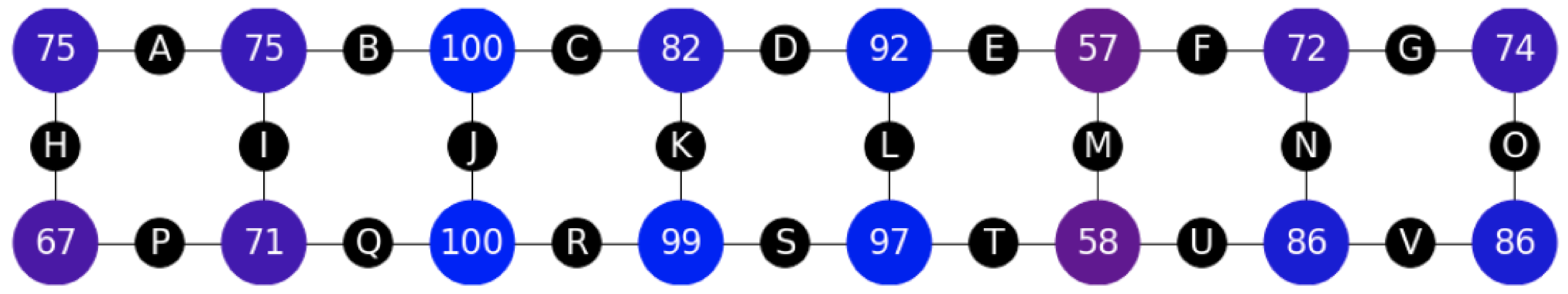
Christian Speicher
27.04.2018, 05.30 Uhr

Merken Drucken Teilen




81 A 61 B 49 C 49 D 85 E 84 F 41 G 88
H I J K L M N O
81 P 62 Q 88 R 88 S 63 T 63 U 44 V 87

In der ersten Runde des Spiels «Quantum Awesomeness» ist es noch einfach, Punktepaare mit ähnlichen Zahlenwerten zu identifizieren.


75 A 75 B 100 C 82 D 92 E 57 F 72 G 74
H I J K L M N O
67 P 71 Q 100 R 99 S 97 T 58 U 86 V 86

In der zweiten Runde wird die Aufgabe bereits schwieriger, weil die Zahlen immer mehr voneinander abweichen. Die Zahl der Punkte und die Verbindungen zwischen ihnen spiegeln die Architektur des Quantenprozessors wider. In diesem Fall handelt es sich um den 16-Bit-Prozessor von IBM. (Bilder: James Wootton, Uni Basel)

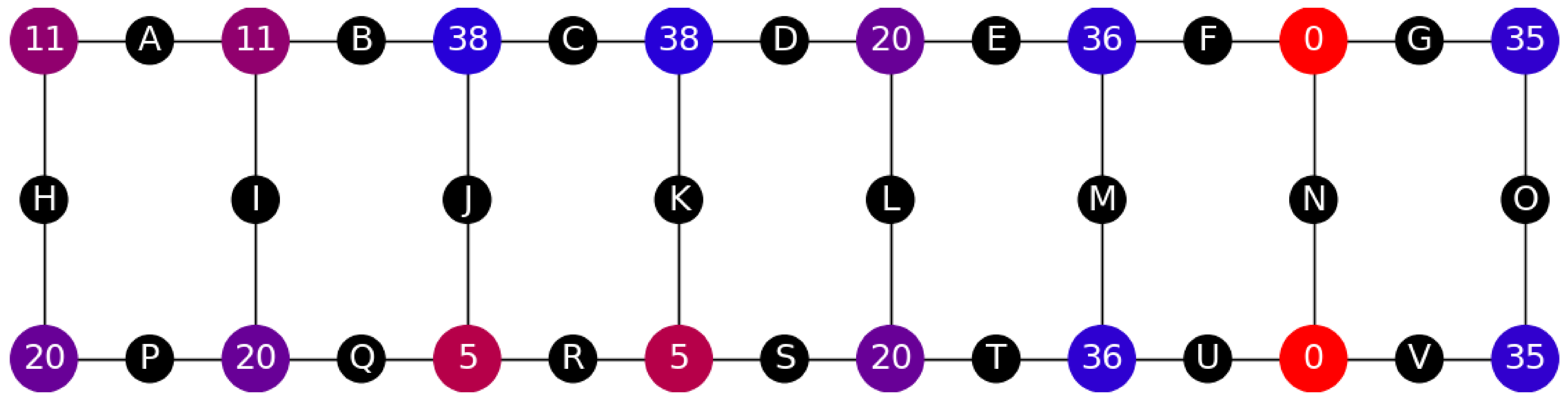
Puzzles on a coupling graph

- We will play a game on a quantum device
  - A random disjoint set of pairs is selected from the coupling graph
  - Each pair is given a random number, displayed on each qubit

- Aim of the game: look at the numbers and figure out the pairing

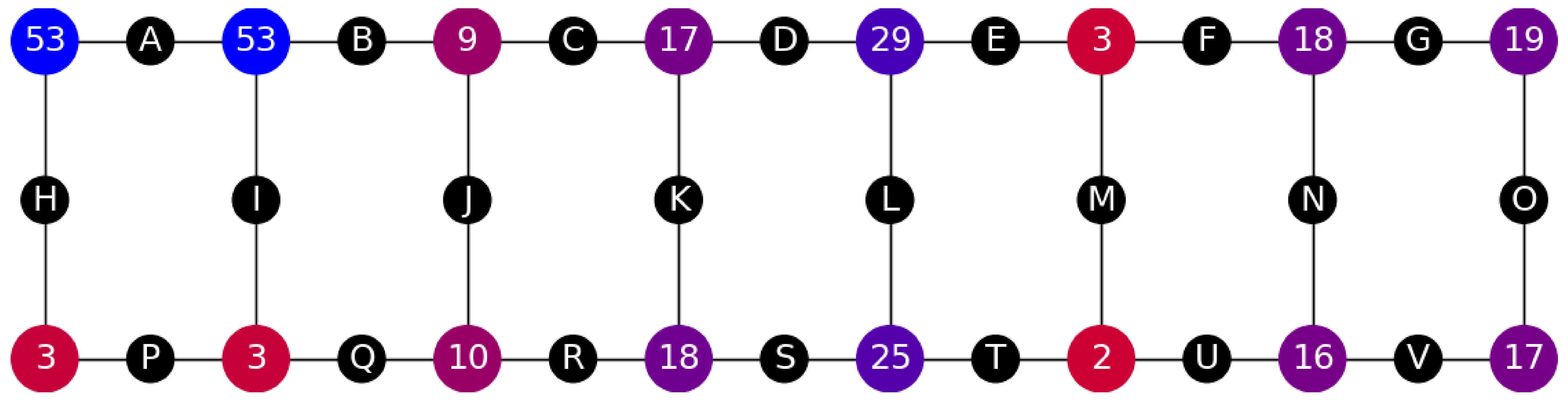
- In later rounds, the numbers get perturbed to increase the difficulty

Round 1



11 A 11 B 38 C 38 D 20 E 36 F 0 G 35
H I J K L M N O
20 P 20 Q 5 R 5 S 20 T 36 U 0 V 35

Round 4



53 A 53 B 9 C 17 D 29 E 3 F 18 G 19
H I J K L M N O
3 P 3 Q 10 R 18 S 25 T 2 U 16 V 17

Circuit for round 1

- Numbers are generated by a quantum circuit
    - For each pair we choose a random angle and entangle them as follows

```
qc.cx(j,k)
qc.rx(theta,j)
qc.cx(j,k)
```

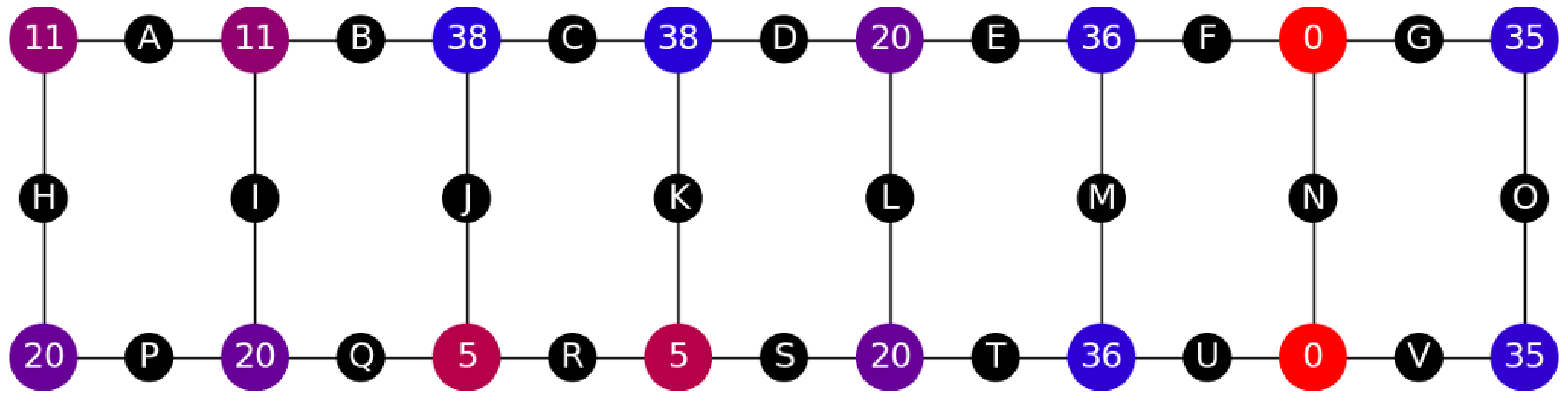
$$\cos\frac{\theta}{2}|00\rangle - i\sin\frac{\theta}{2}|11\rangle$$

- The probability of a **1** outcome on each qubit is then

$$p_j = p_k = \sin^2\frac{\theta}{2}$$

- This is the same for the two qubits of each pair, but varies from pair to pair

- The probability of a **1** is then the number reported



Circuit for round 2+

- In theory:
  - First remake round 1:

```
qc.cx(j,k)
qc.rx(theta,j)
qc.cx(j,k)
```

  - Then unmake it:

```
qc.cx(j,k)
qc.rx(-theta,j)
qc.cx(j,k)
```

  - Then generate a new random pairing and entangle those pairs

- In practice:
  - For the pairs of round 1, apply

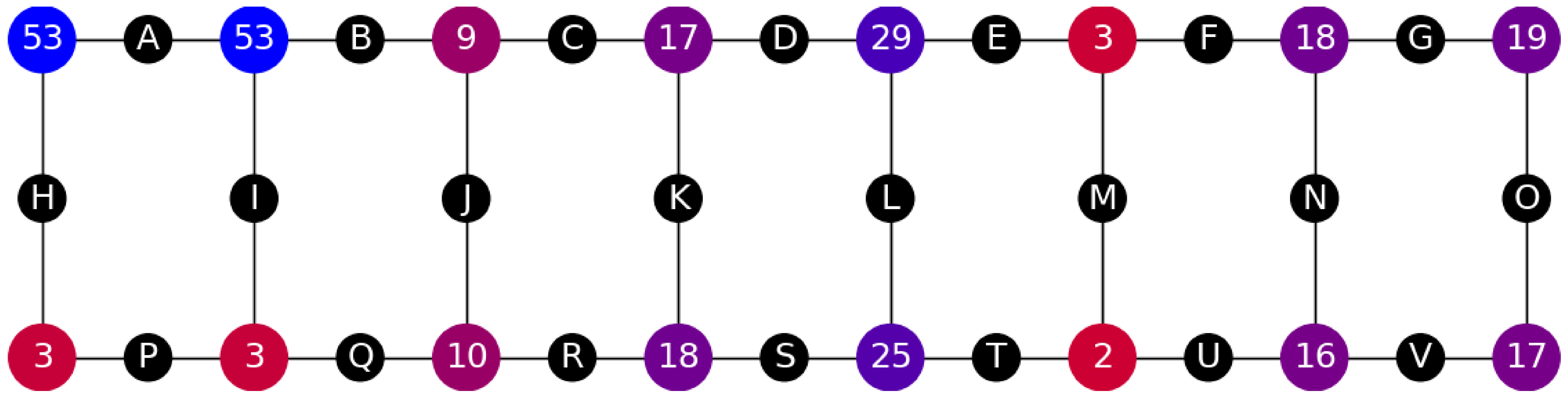
```
qc.cx(j,k)
qc.u(0,-pi/2,pi)
qc.cx(j,k)
```

  (which is equivalent to the identity)

  - Then generate a new random pairing and entangle those pairs

- The remake/unmake rounds are also conjugated by random `rx` or `ry` rotations

- Buildup of errors causes the numbers to drift from their correct values



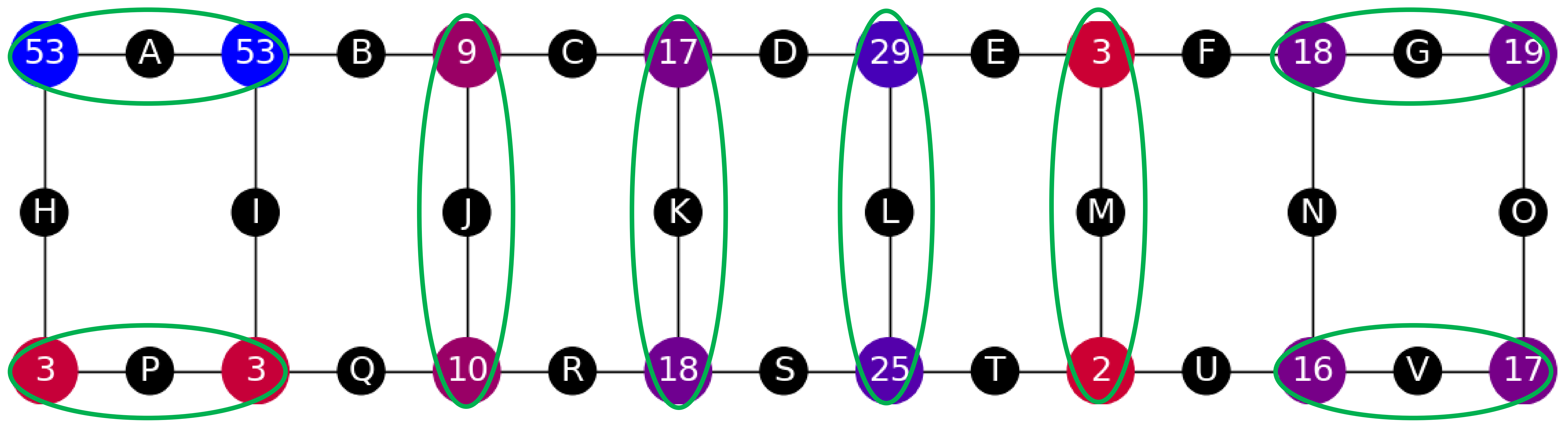
Quantifying awesomeness

- With many samples, how can we quantify quality?

- Fuzz:
  - For the two qubits in each pair, how different are their values? $|p_j - p_k|$

- Difference
  - For each qubit, how different is its value from what it should be? $\left|\frac{2\arcsin\sqrt{p_j}}{\pi} - \theta\right|$

- MWPM correctness:
  - Using an algorithm to guess the pairing, how often is it right?



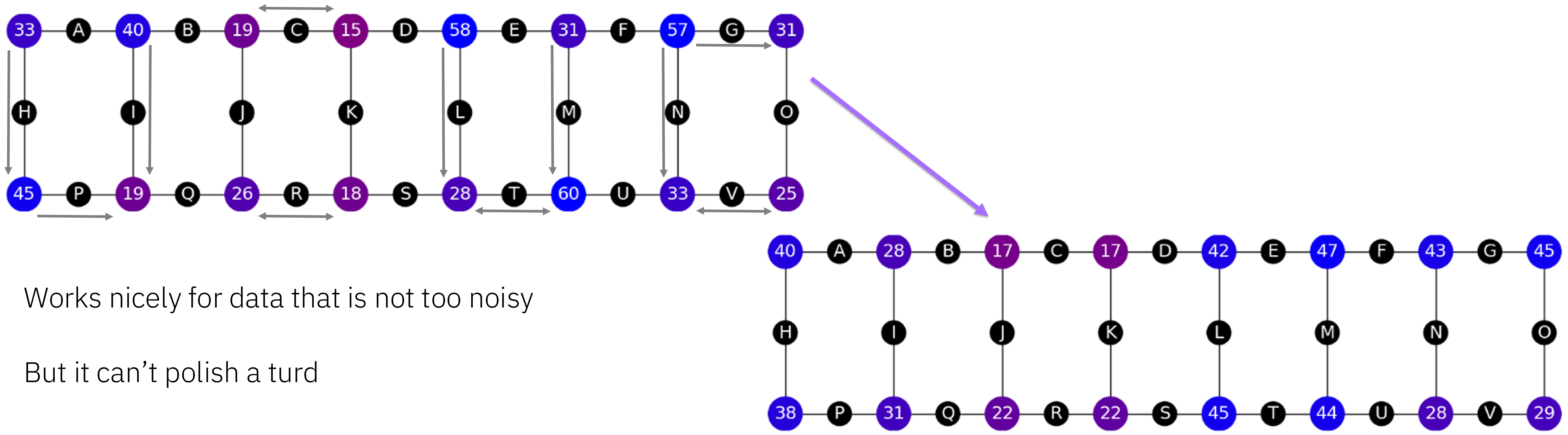
Cleaning up with correlation data

- We also calculate $Z_j Z_k$ for each potential pair
  - The state implies that $Z_j Z_k = 1$ for entangled pairs
  - Noise, or both qubits otherwise maximally entangled, leads to $Z_j Z_k = 0$

- So for each qubit *j* we find *j'*, for which $Z_j Z_k$ is maximal
  - Without noise, we expect $j' = k$

- To clean up the values we then make this assumption:

$$p_j = \frac{p_j + p_{j'}}{2}$$



- Works nicely for data that is not too noisy

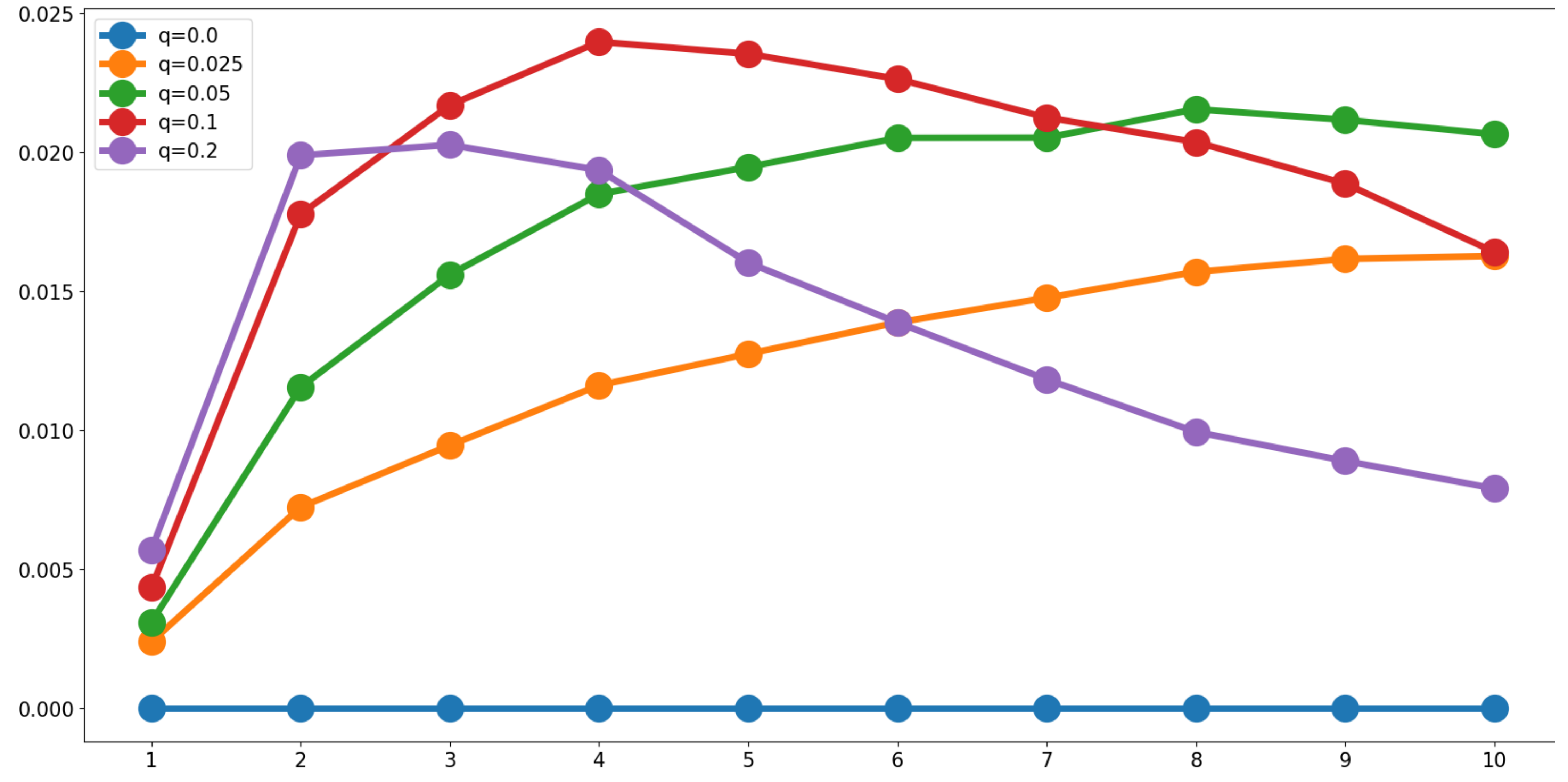
- But it can't polish a turd

Results from noisy simulations

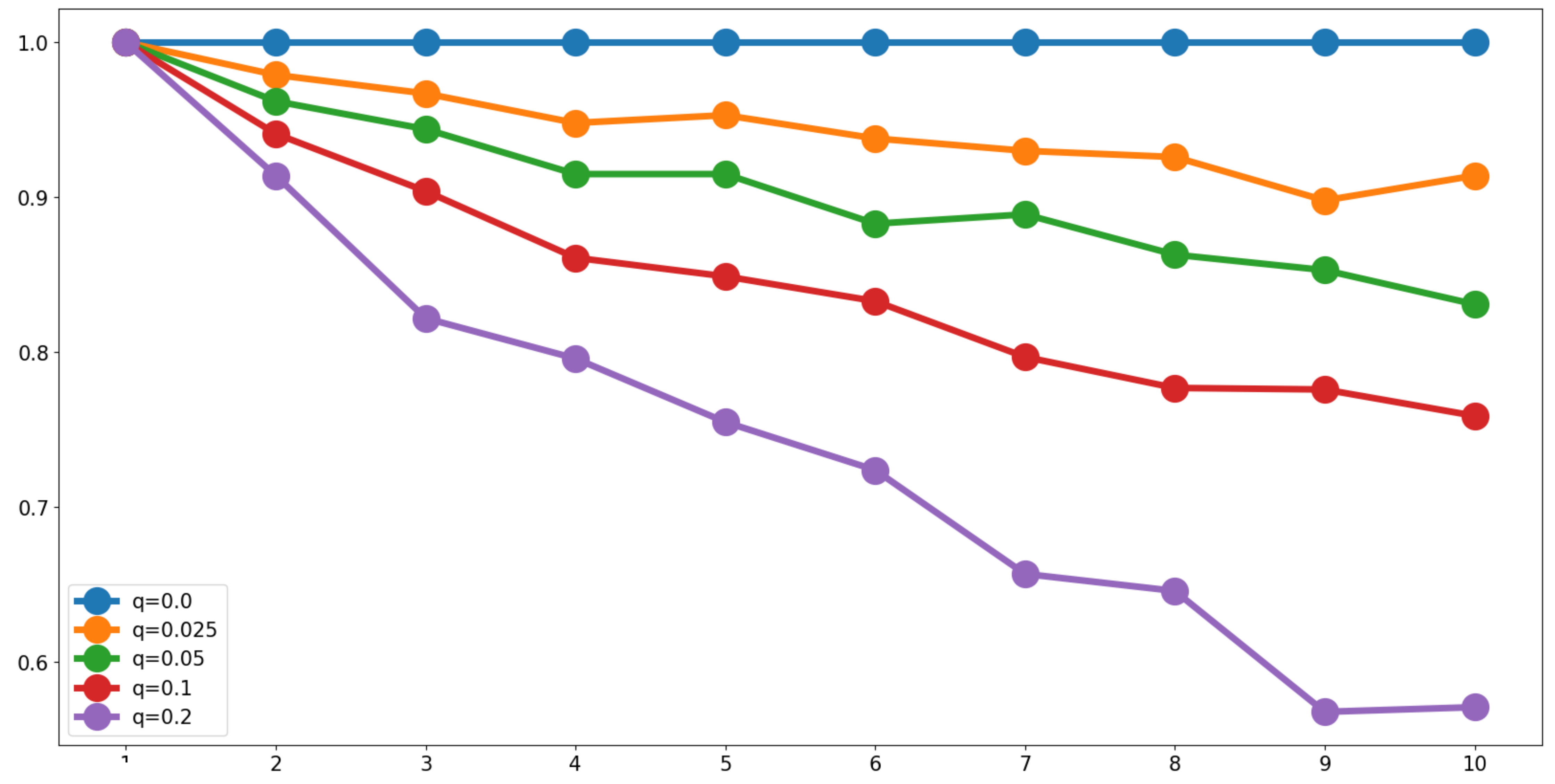
- Simulations of 5 qubit `ibmqx4`
- Depolarizing noise on every gate with probability *p*

- For the fuzziness we get the 'peak of doom'
  - Low fuzziness at the beginning because all is well
  - Low fuzziness at the end because everything is awful
  - In the middle we get a landmark

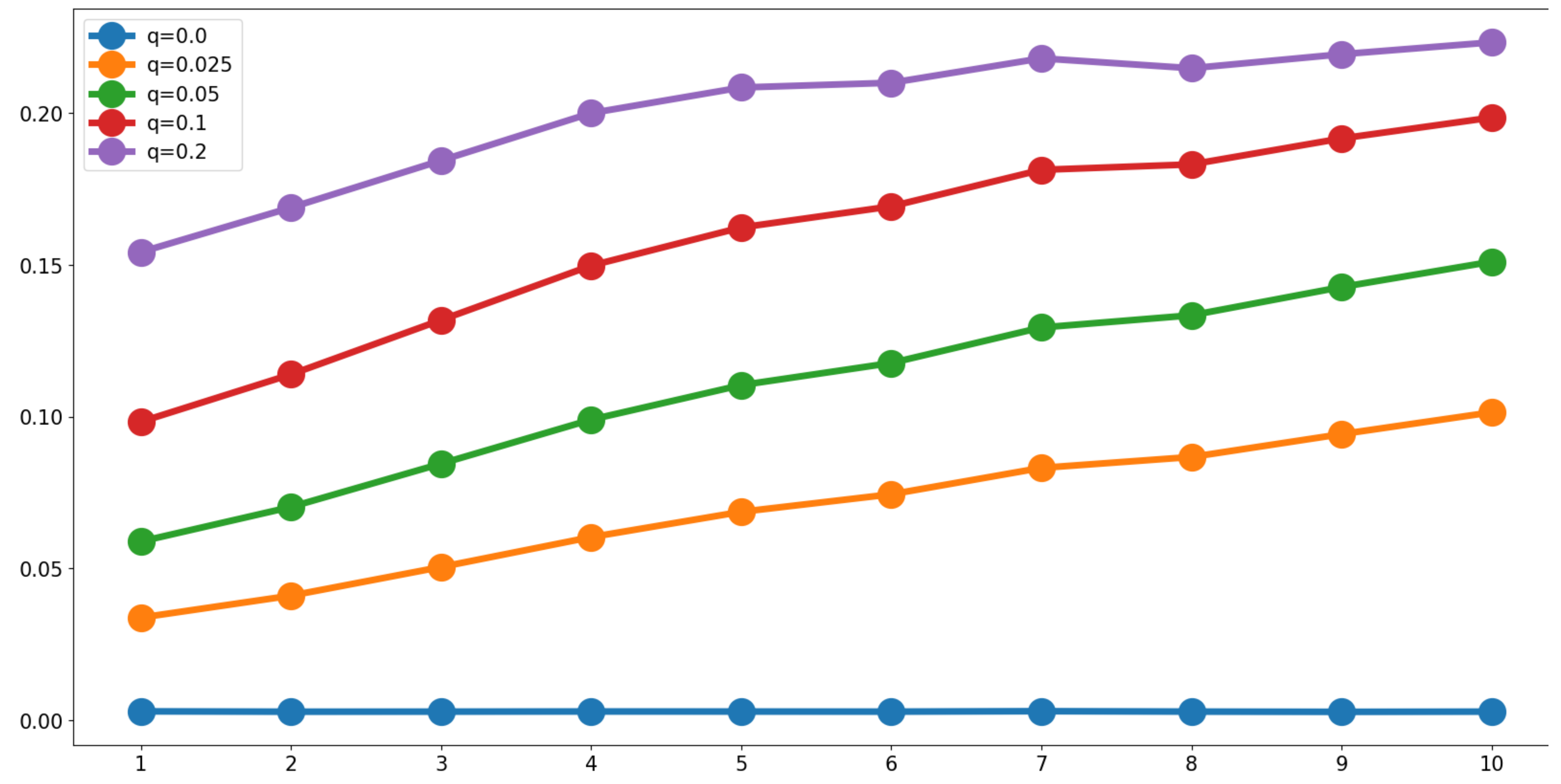
Fuzziness



Matching success



Difference

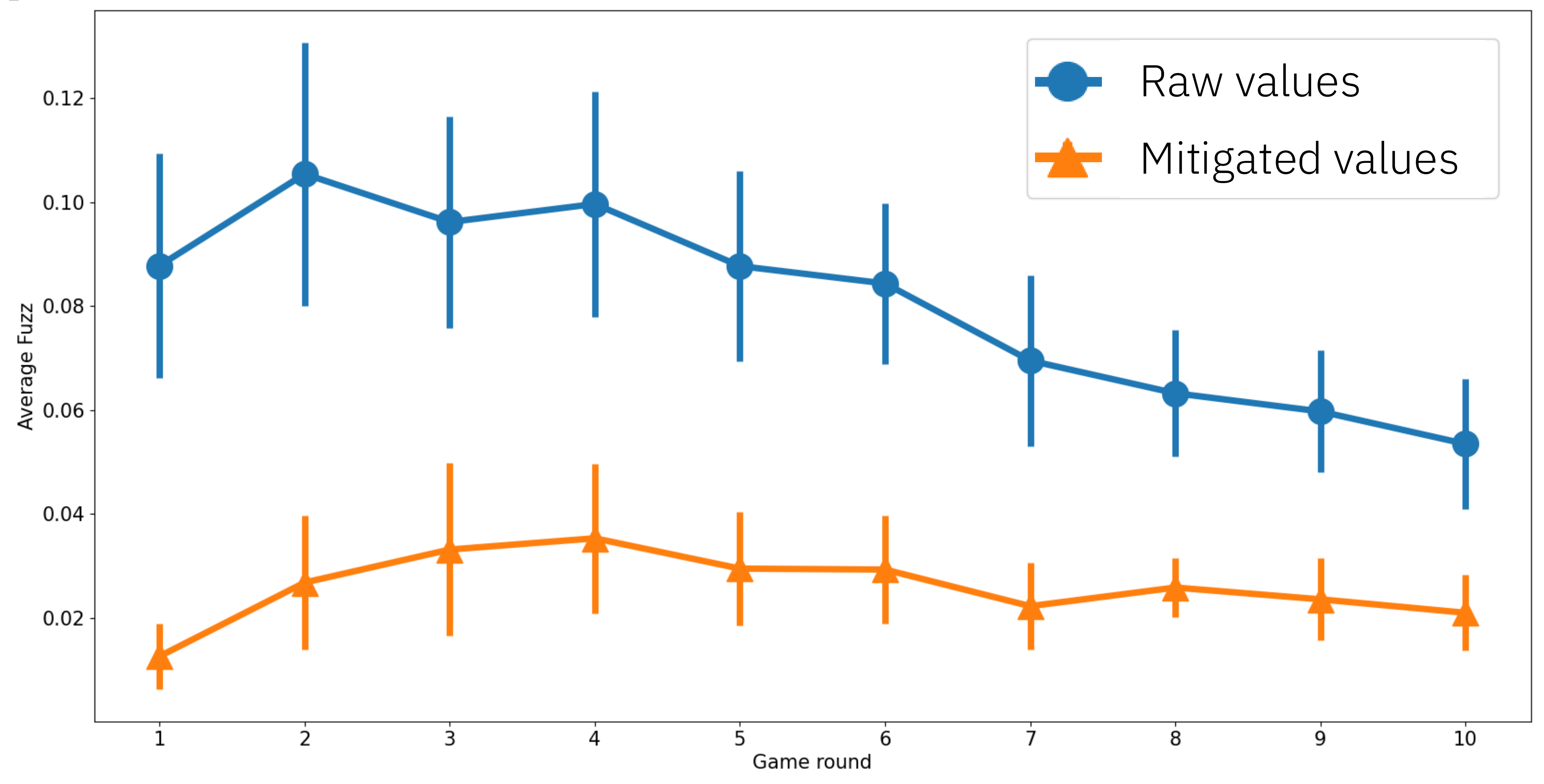


Results from `ibm_rueschlikon`

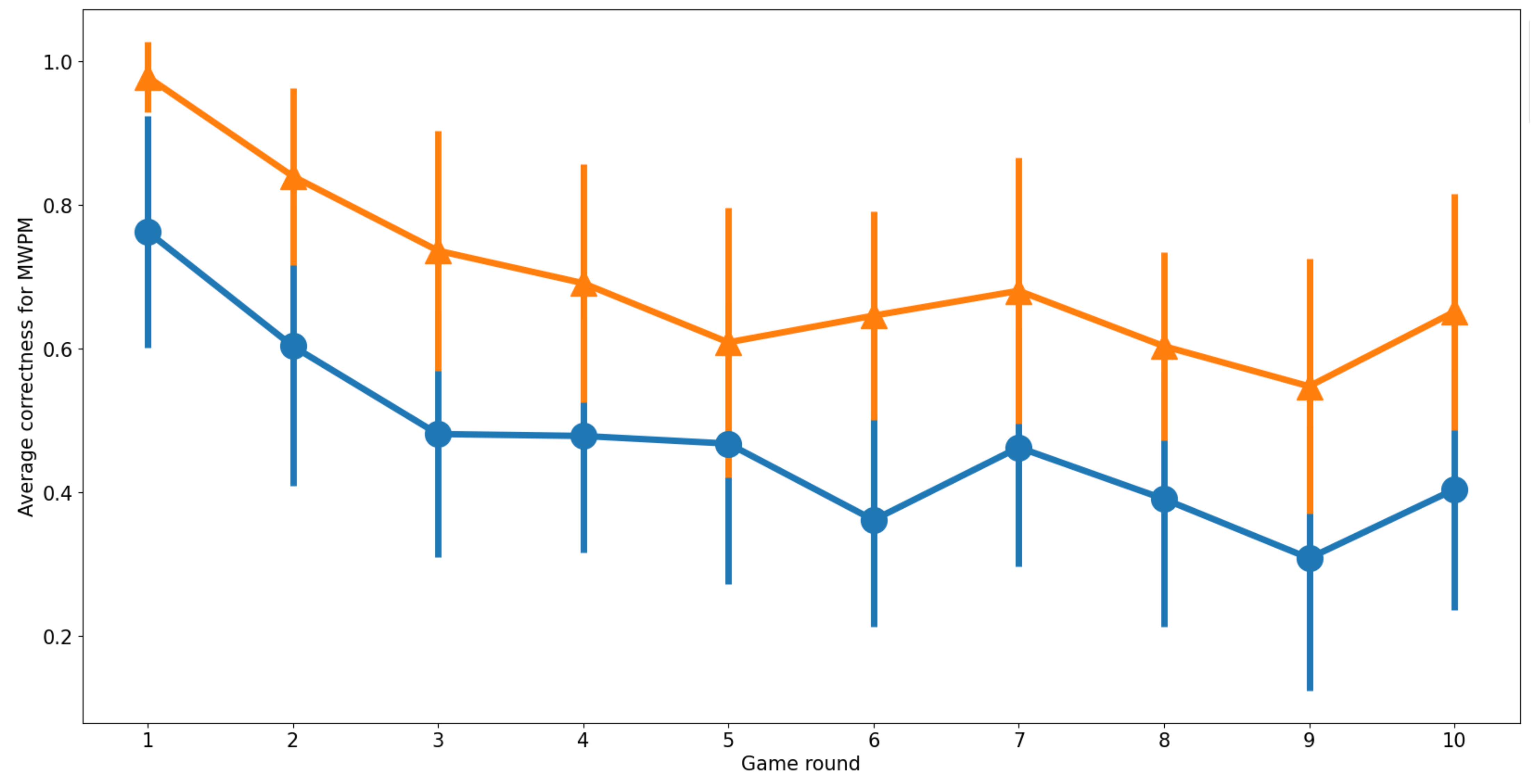
- 16 qubit device (also known as `ibmqx5`)

- Peak of doom already at ~round 2
  - CX depth of 4
- Matching success and difference converged by ~round 10
- Great for its time, but not really very good

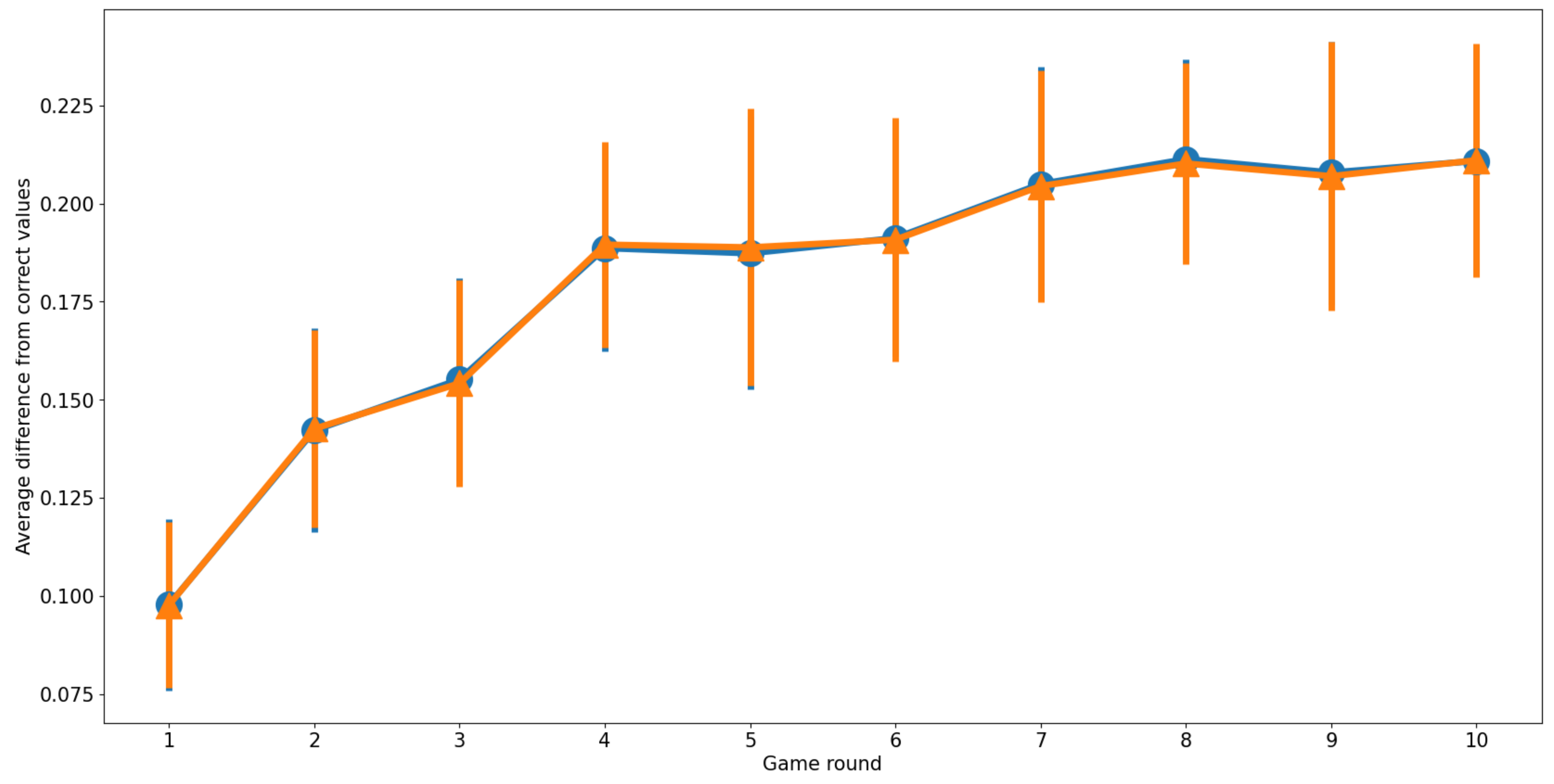
Fuzziness



Matching success



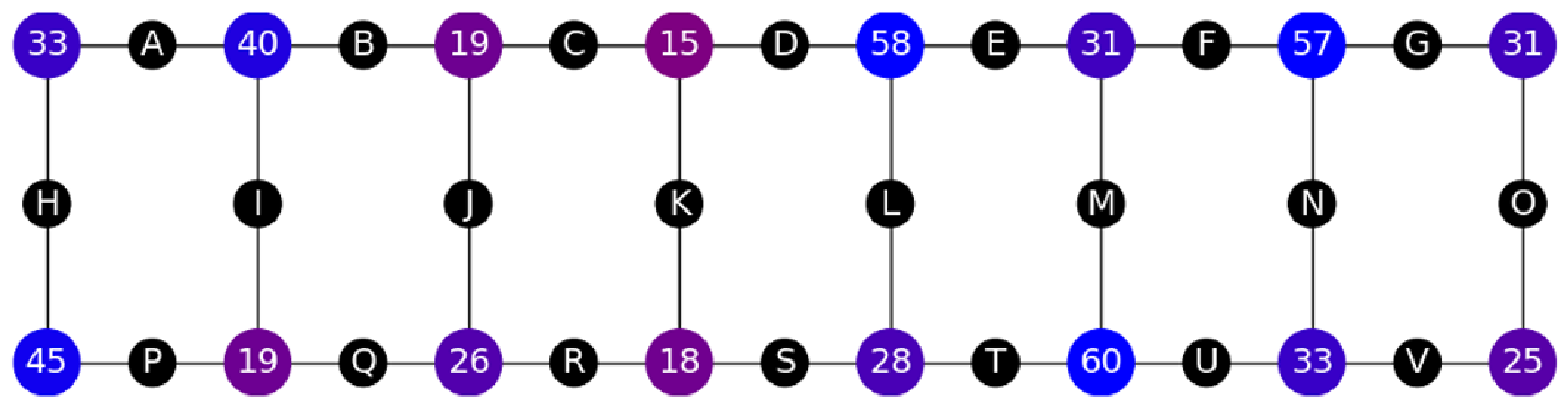
Difference



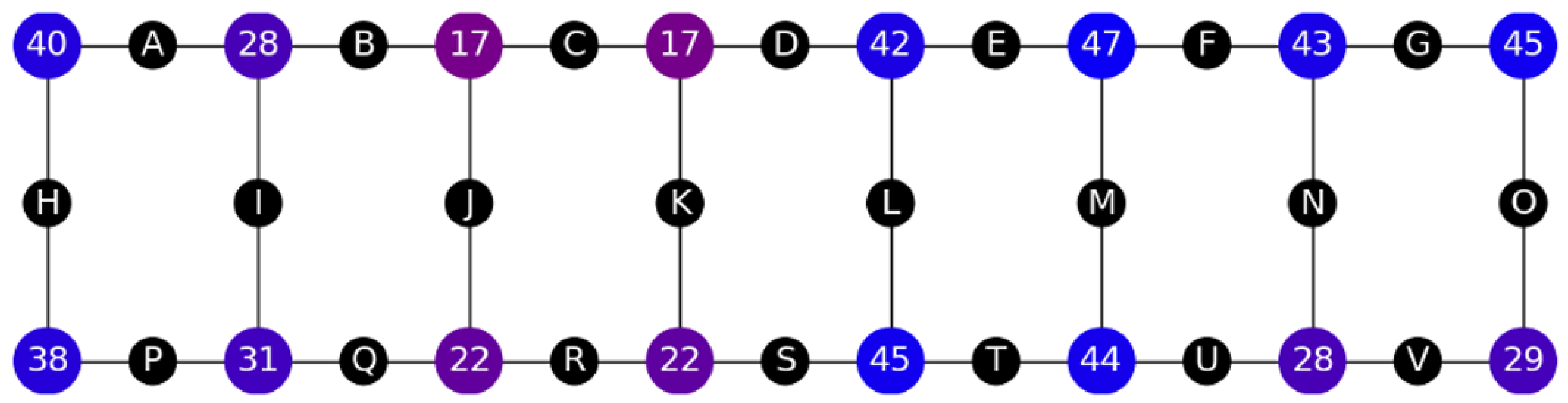
Puzzles from `ibm_rueschlikon`

- This is also seen from the puzzles. Here's an attempt at round 2
- Correct solution: C, H, I, R, L, M, N, O
- Mitigation helps a bit
- But it still doesn't give very good results

Round 2



Round 2 (mitigated)

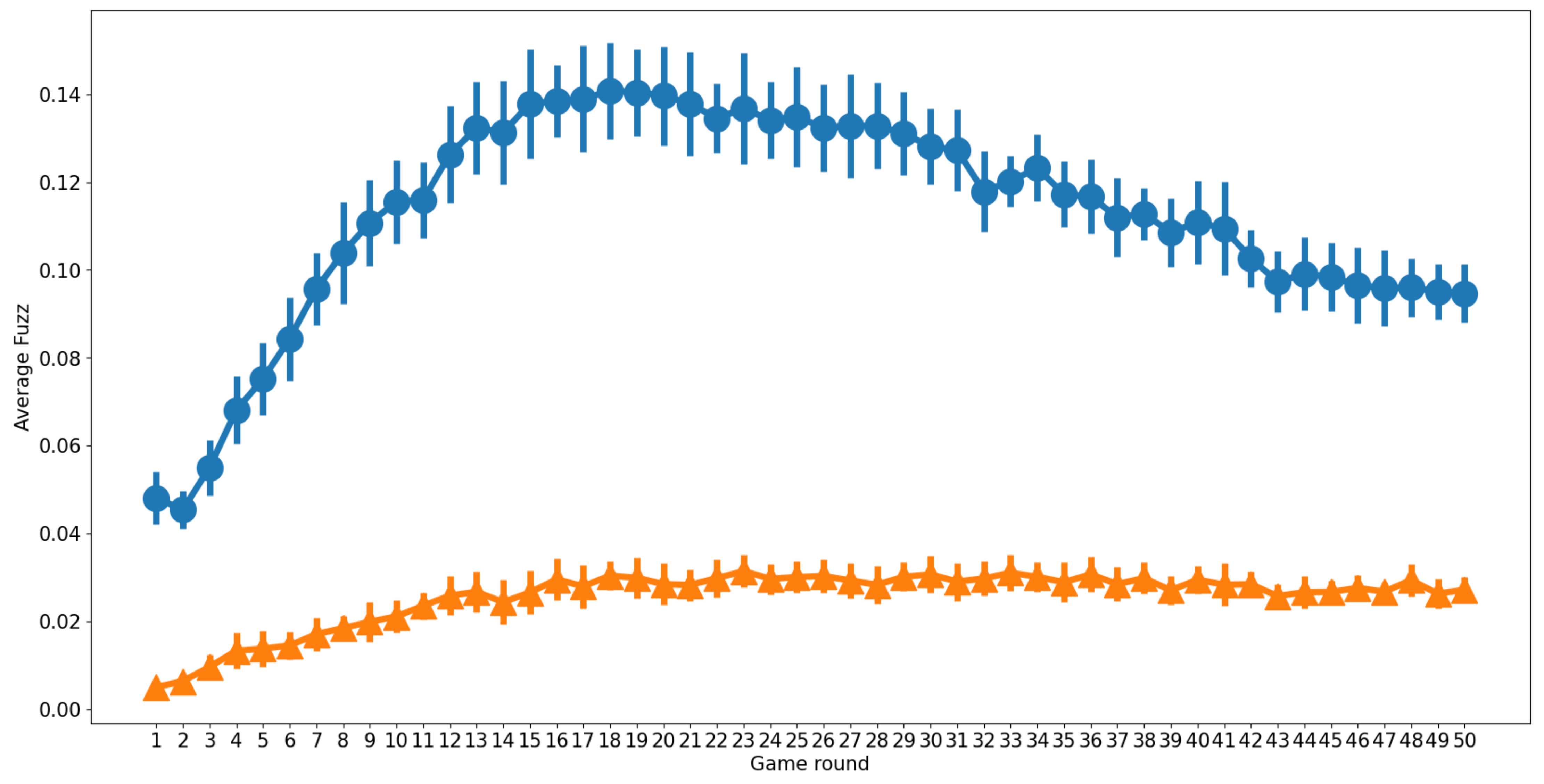


Results from `ibm_torino`

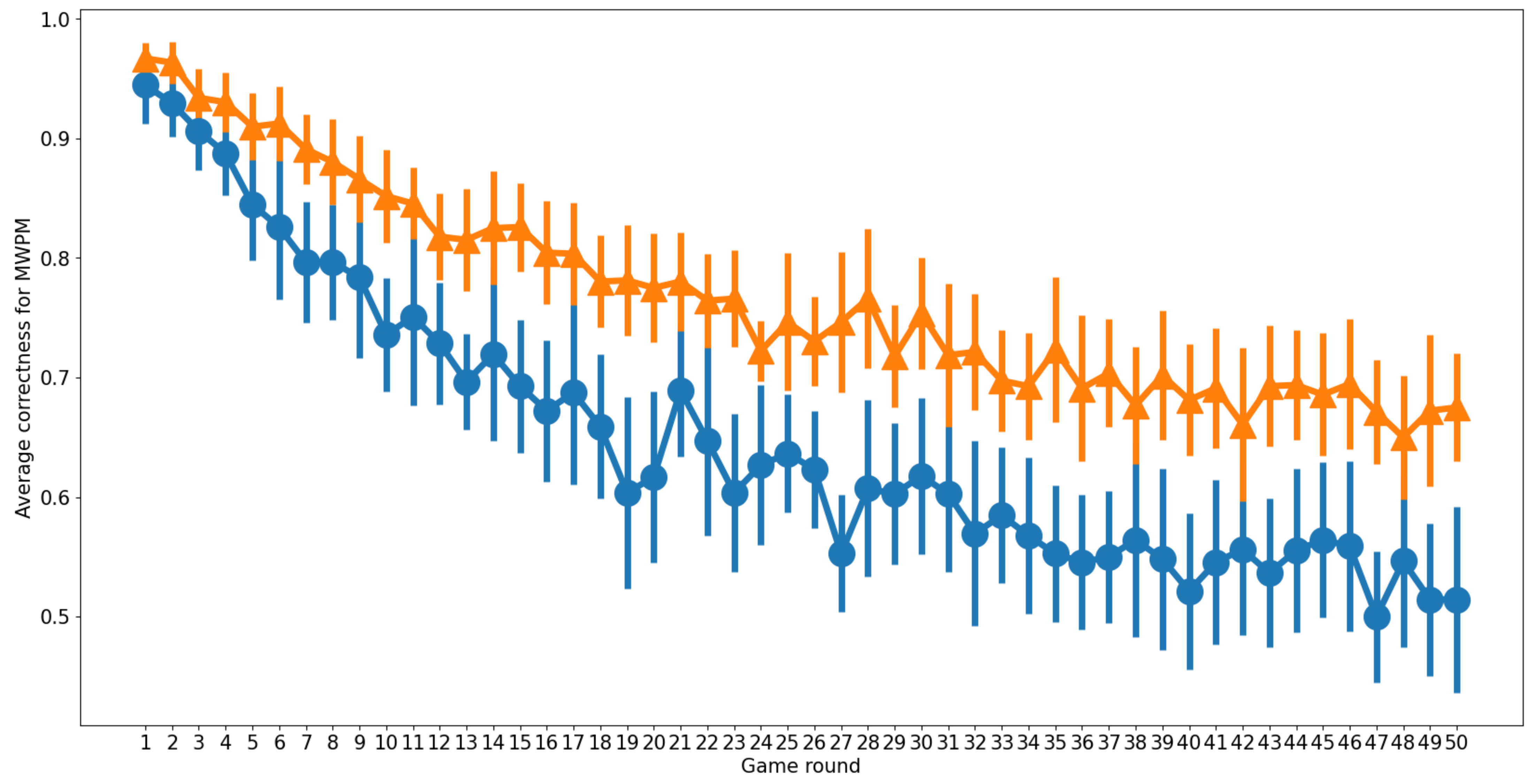
- 133 qubit *Heron* device
- Best ELPG at time of running

- Matching success and difference starting to converge
- Much better than `ibmq_rueschlikon`

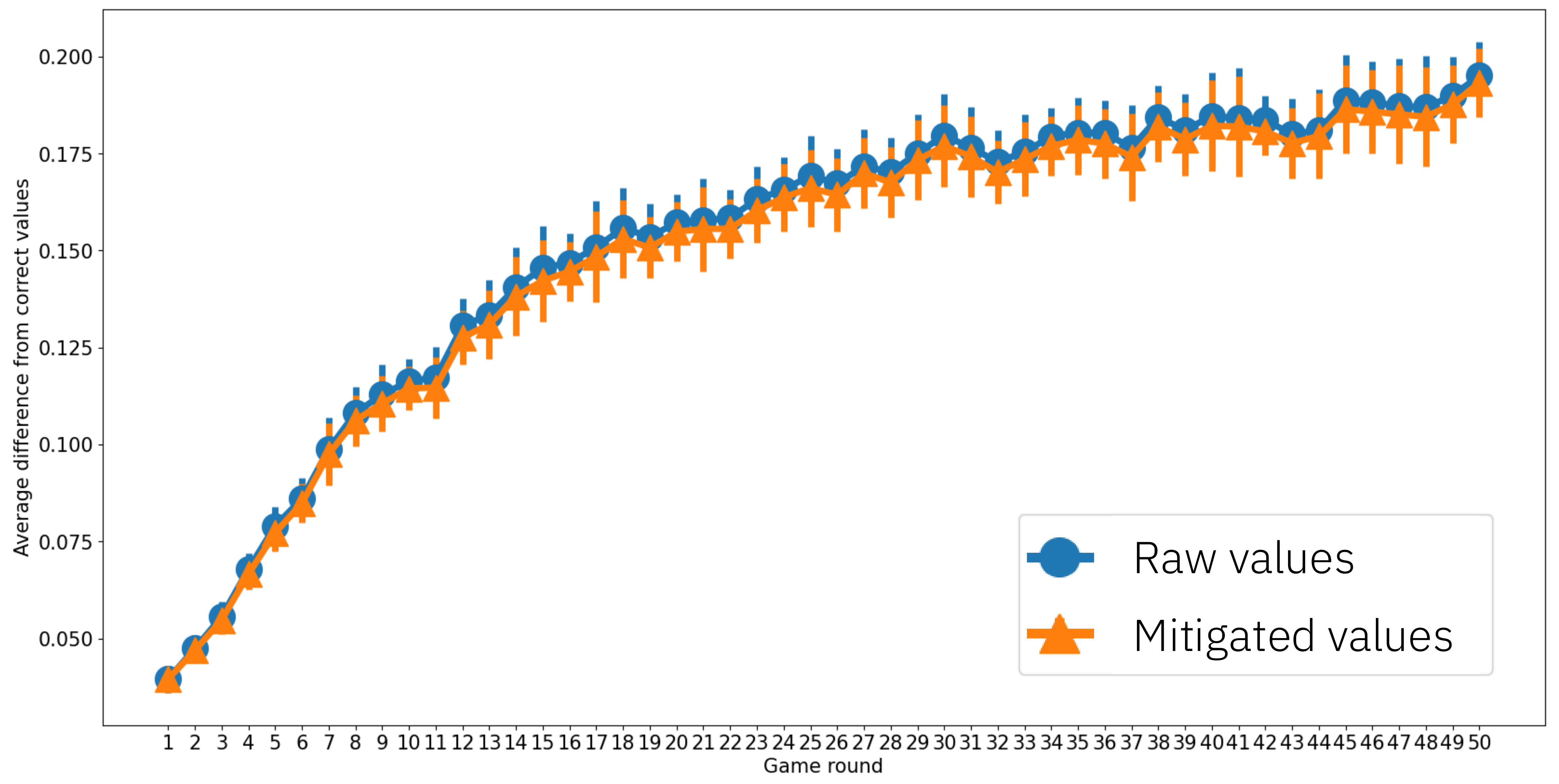
Fuzziness



Matching success



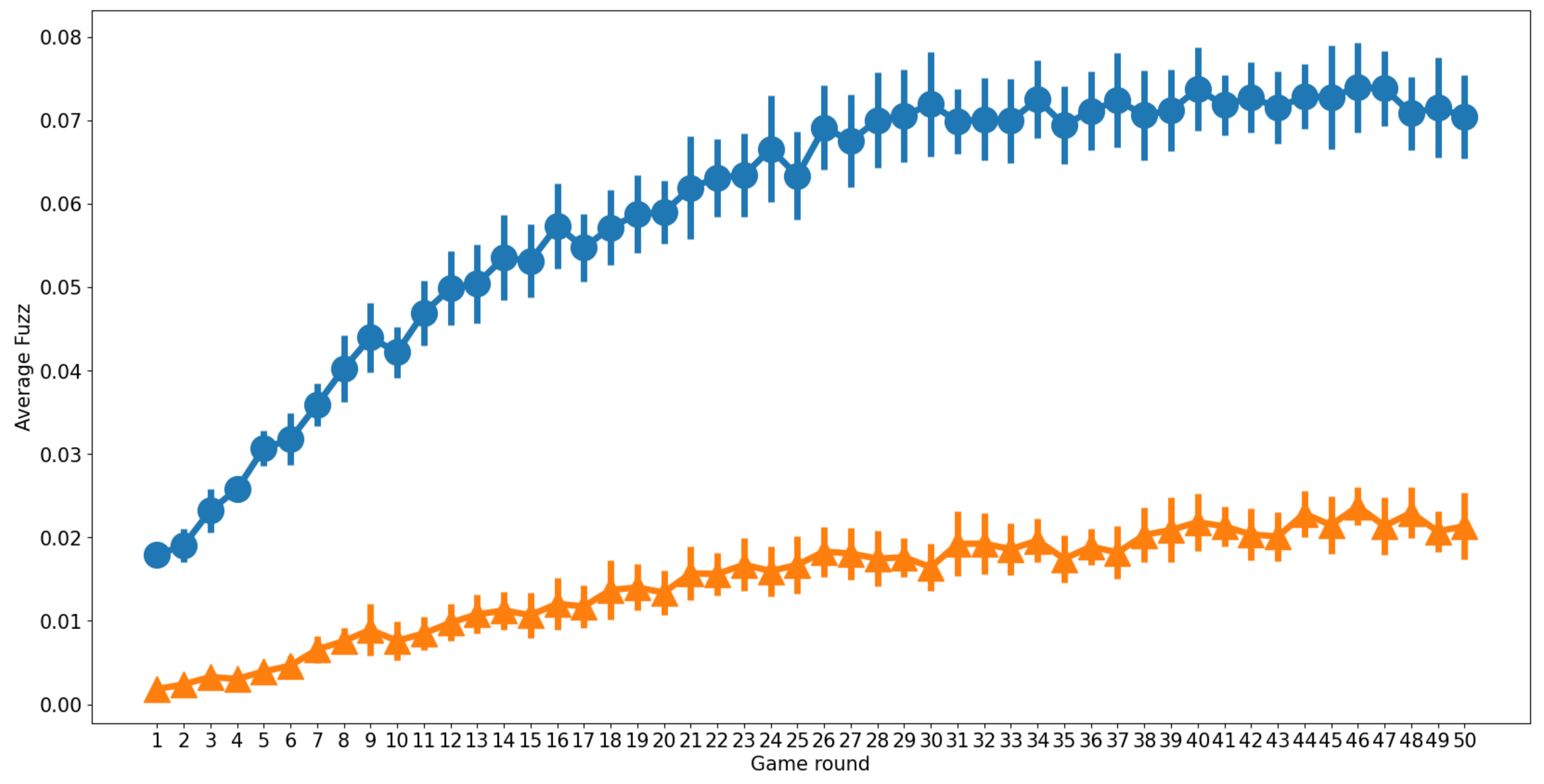
Difference



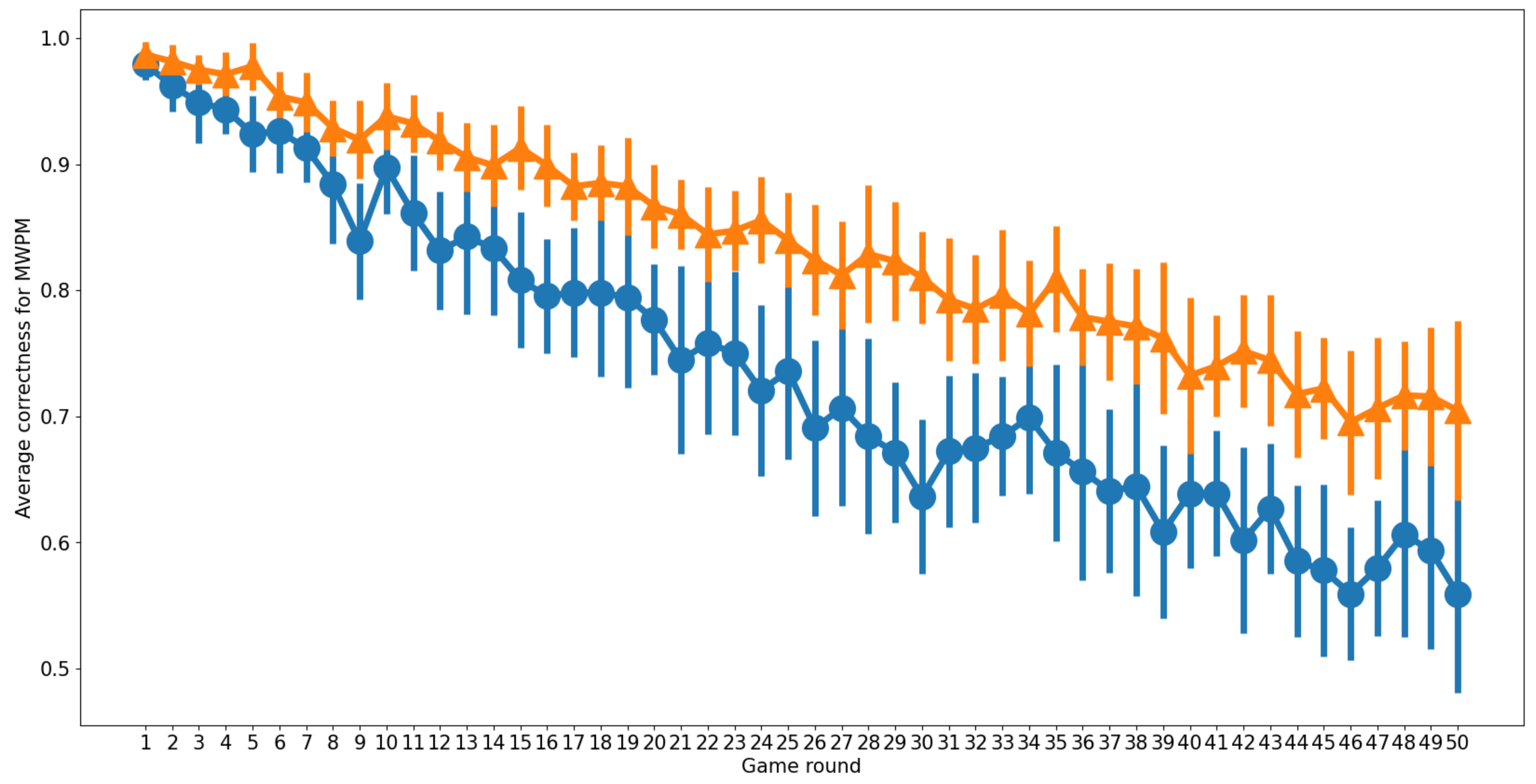
Results from `ibm_kyiv`

- 127 qubit *Eagle* device
- Peak of doom pushed out to around round 50
  - CX depth of ~100
- Matching success still at 70% for round 50 (would be ~40% for a random solution)

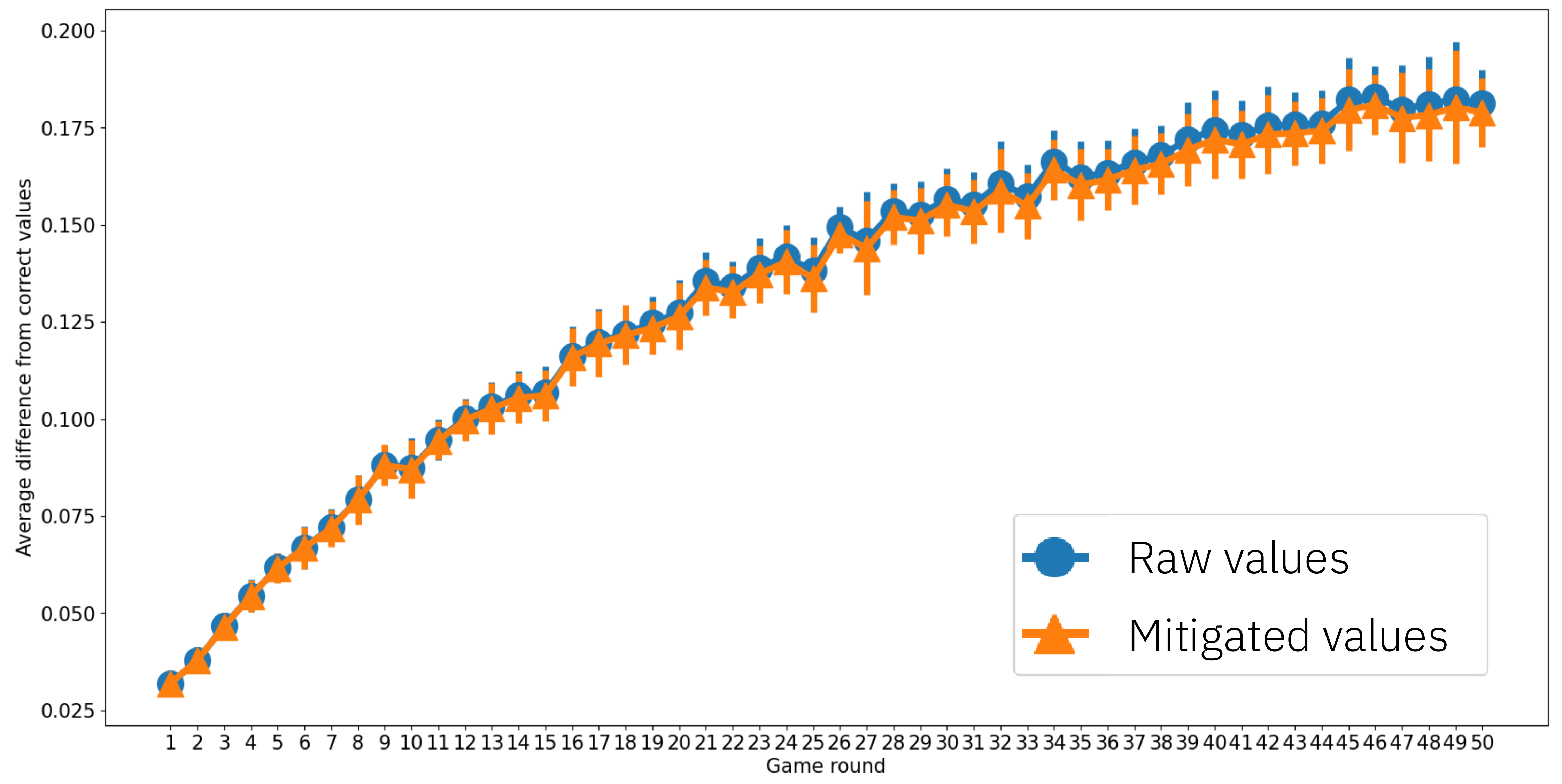
Fuzziness



Matching success



Difference

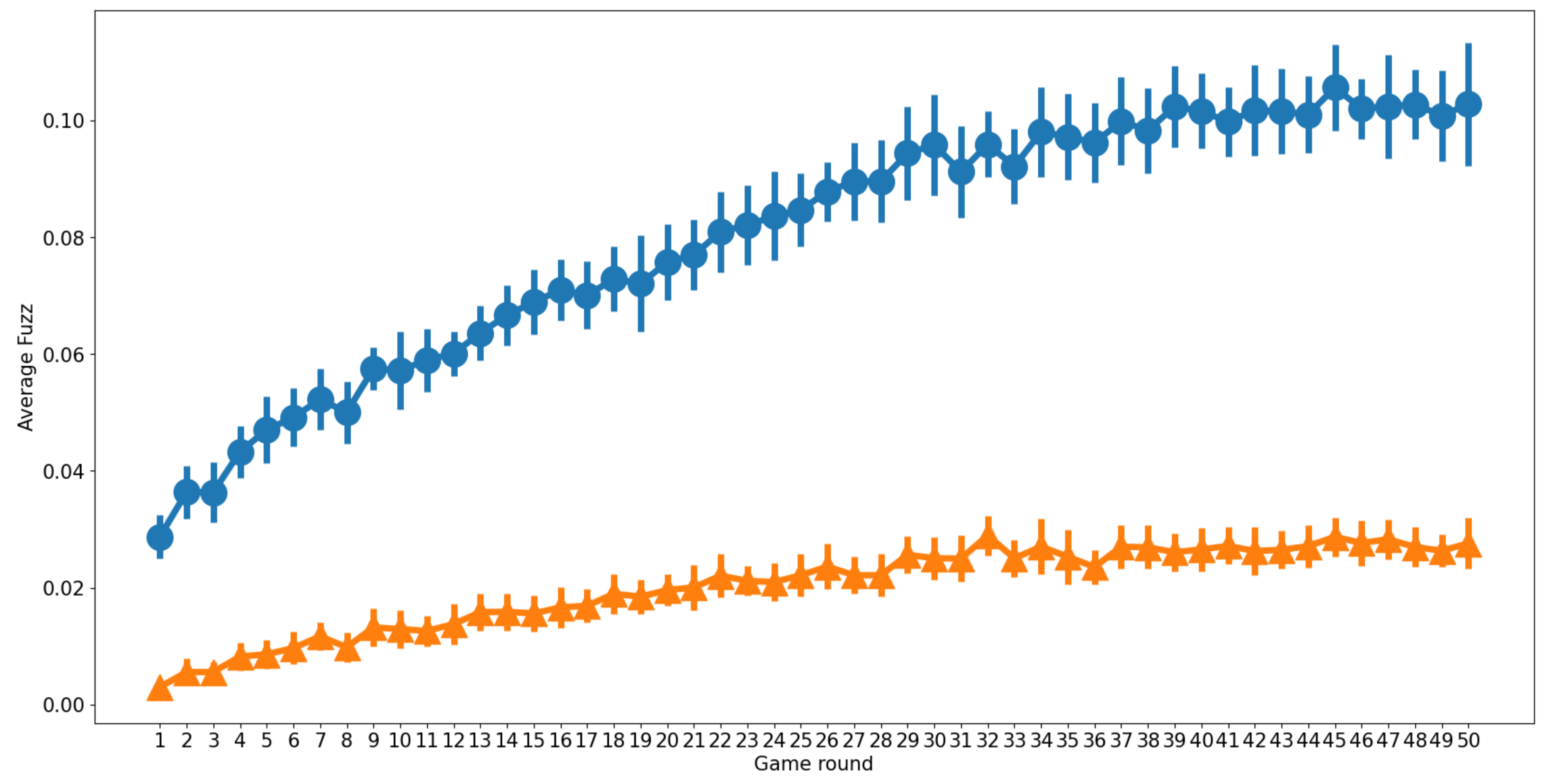


Results from `ibm_fez`

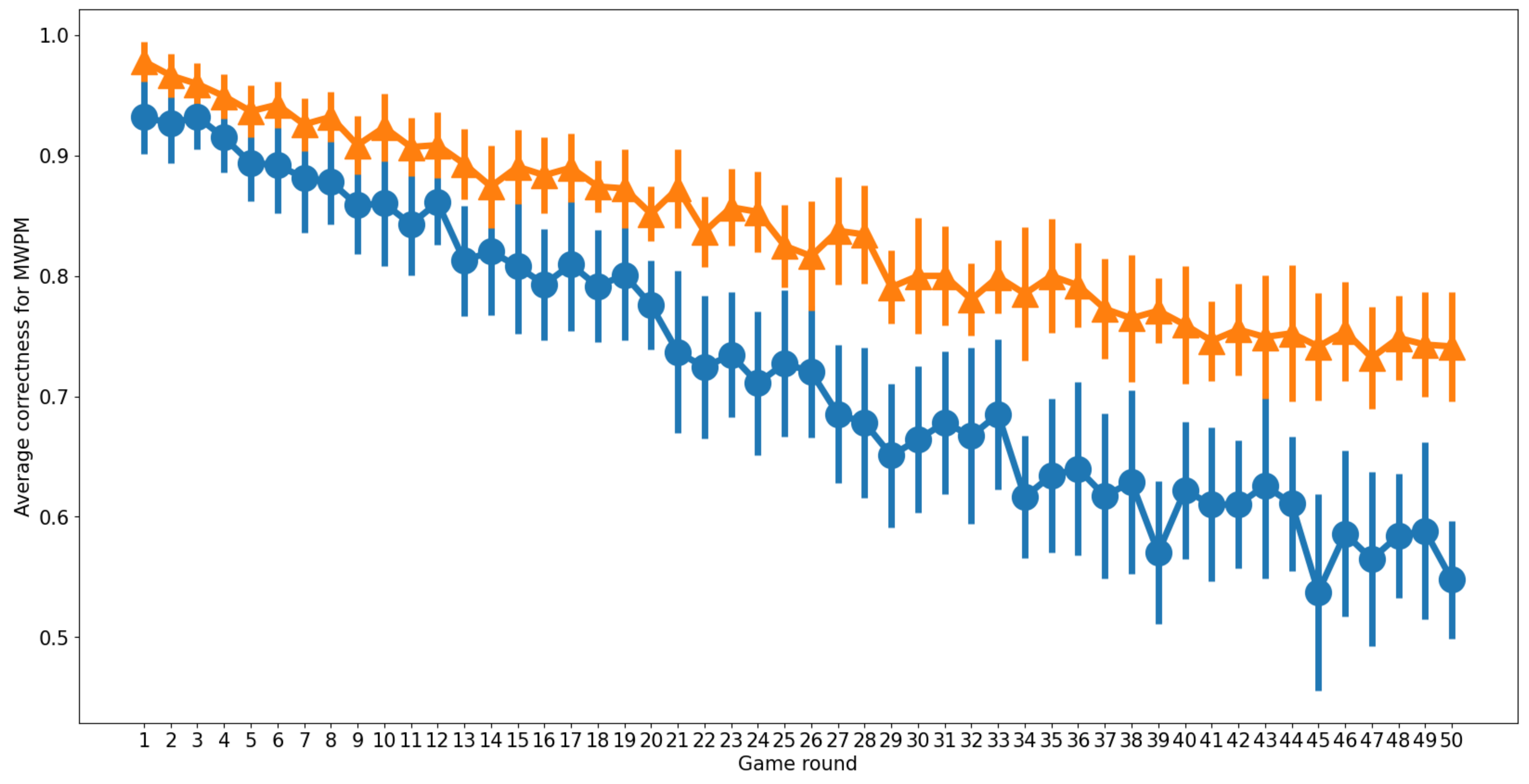
- 156 qubit *Heron* device
- Better ELPG than `ibm_torino`

- Similar behaviour to `ibm_kyiv`, but with more qubits

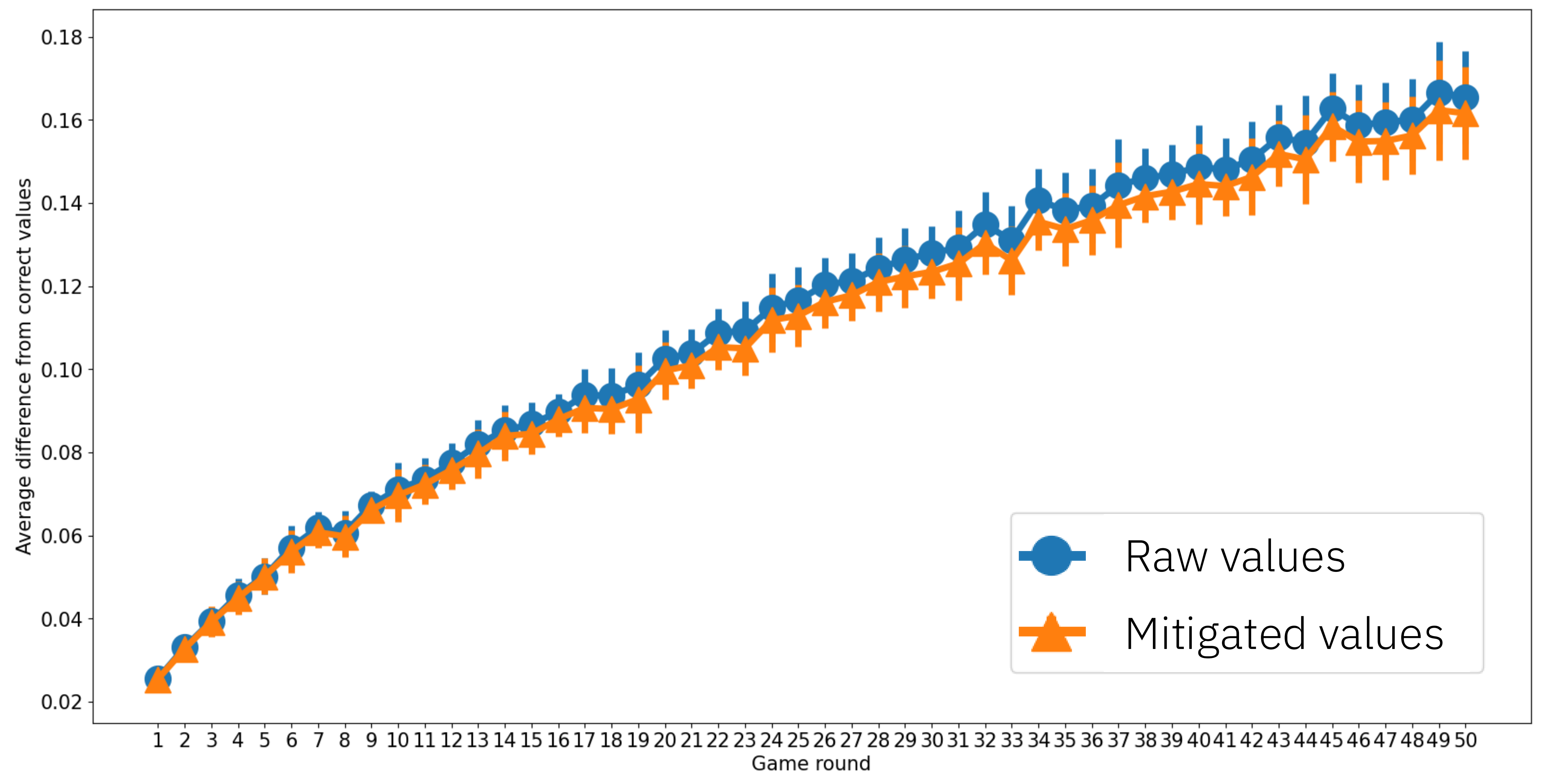
Fuzziness



Matching success

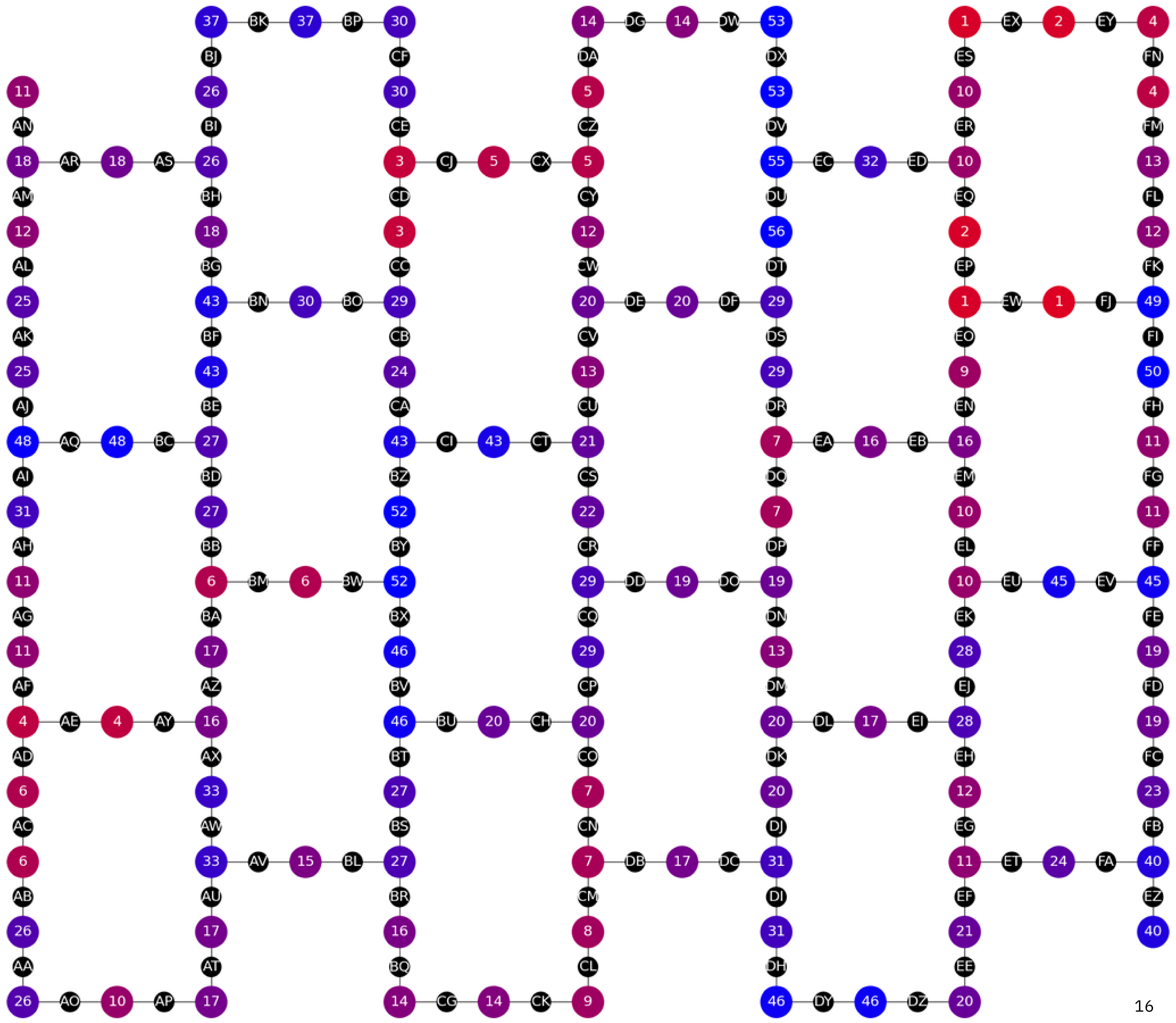


Difference



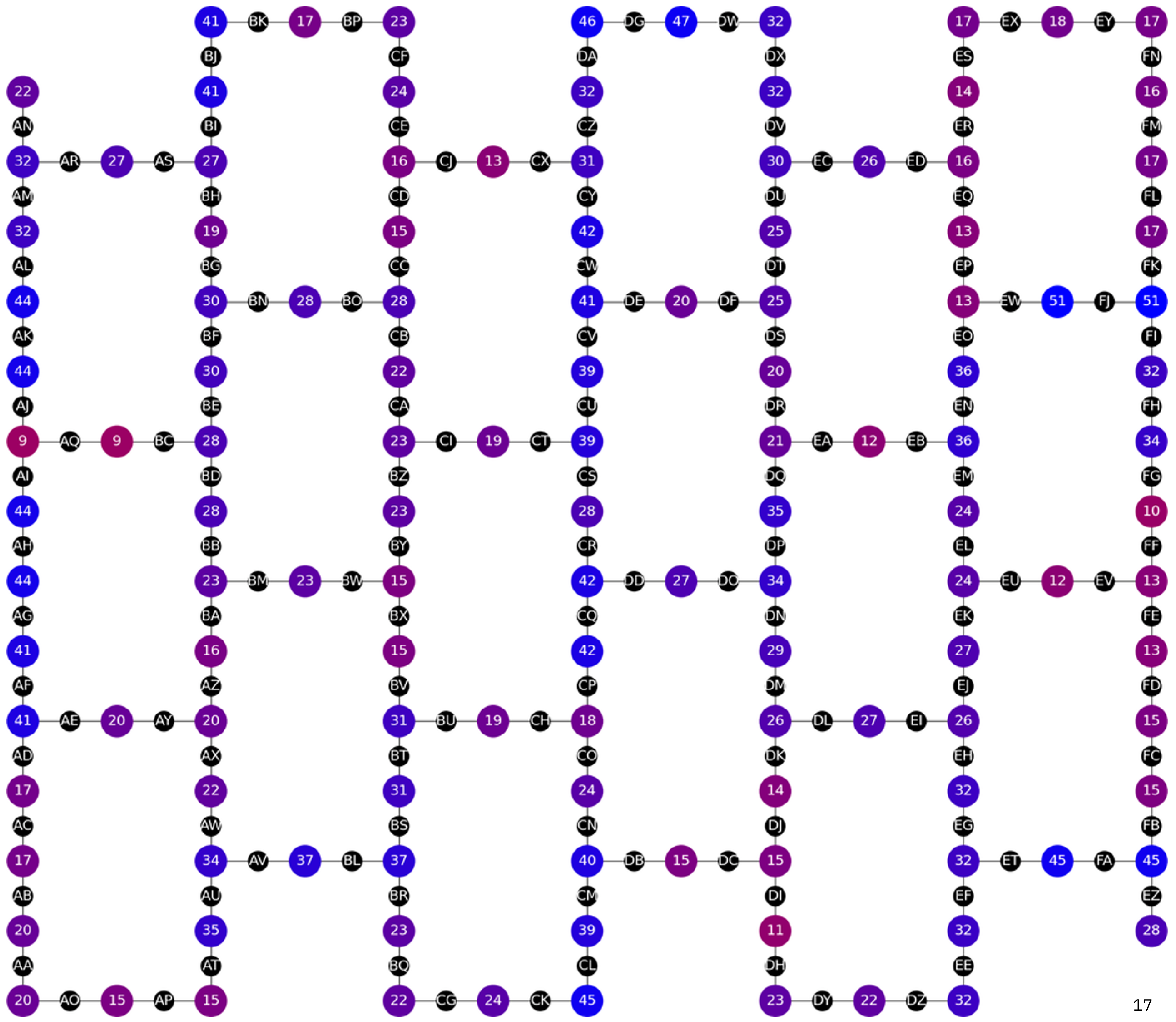
Puzzles from `ibm_kyiv`

- Example of a round 1 puzzle with cleanup
  Note: in this coupling graph, some qubits are left unpaired



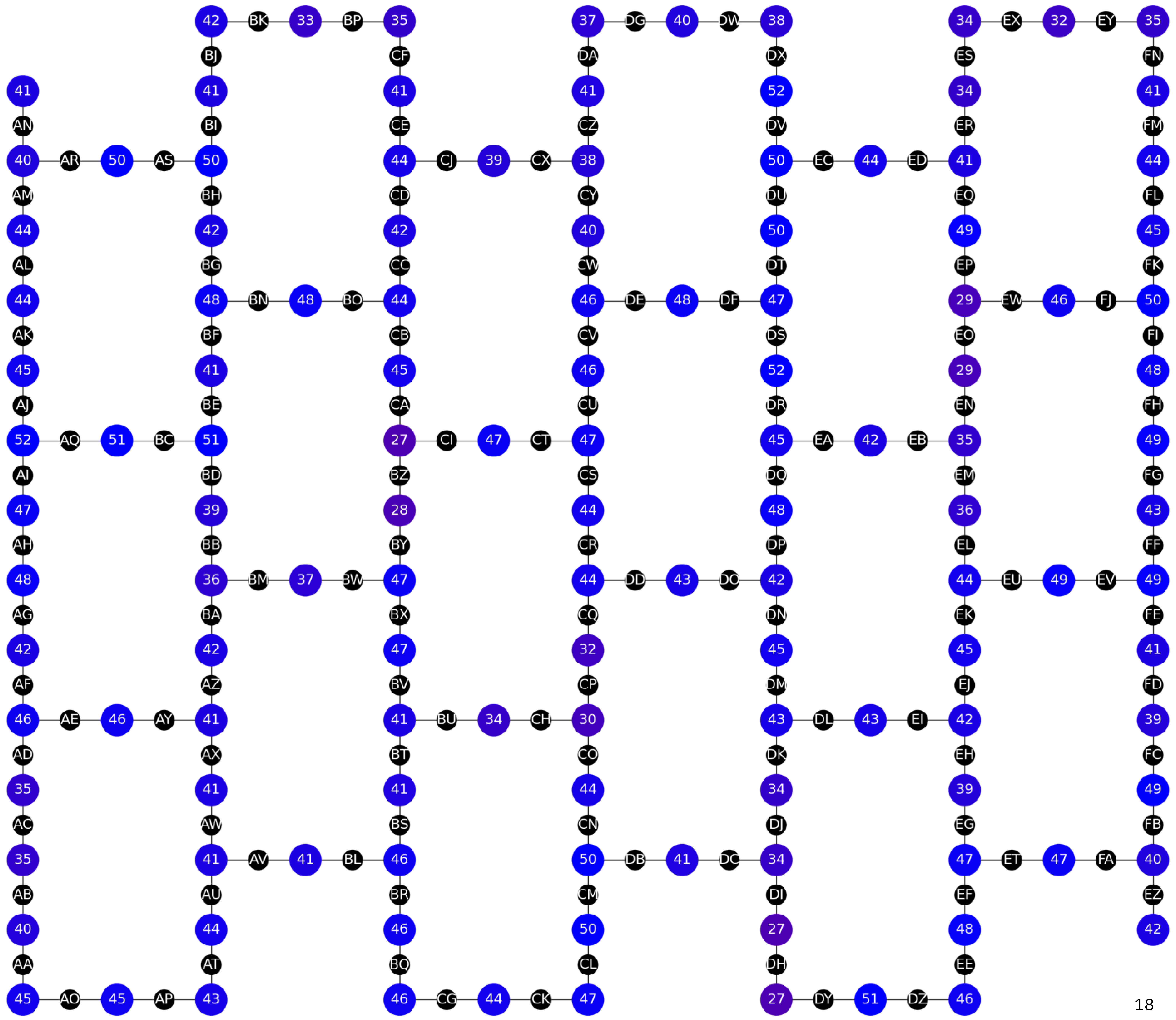
Puzzles from `ibm_kyiv`

- Example of a round 10 puzzle with cleanup



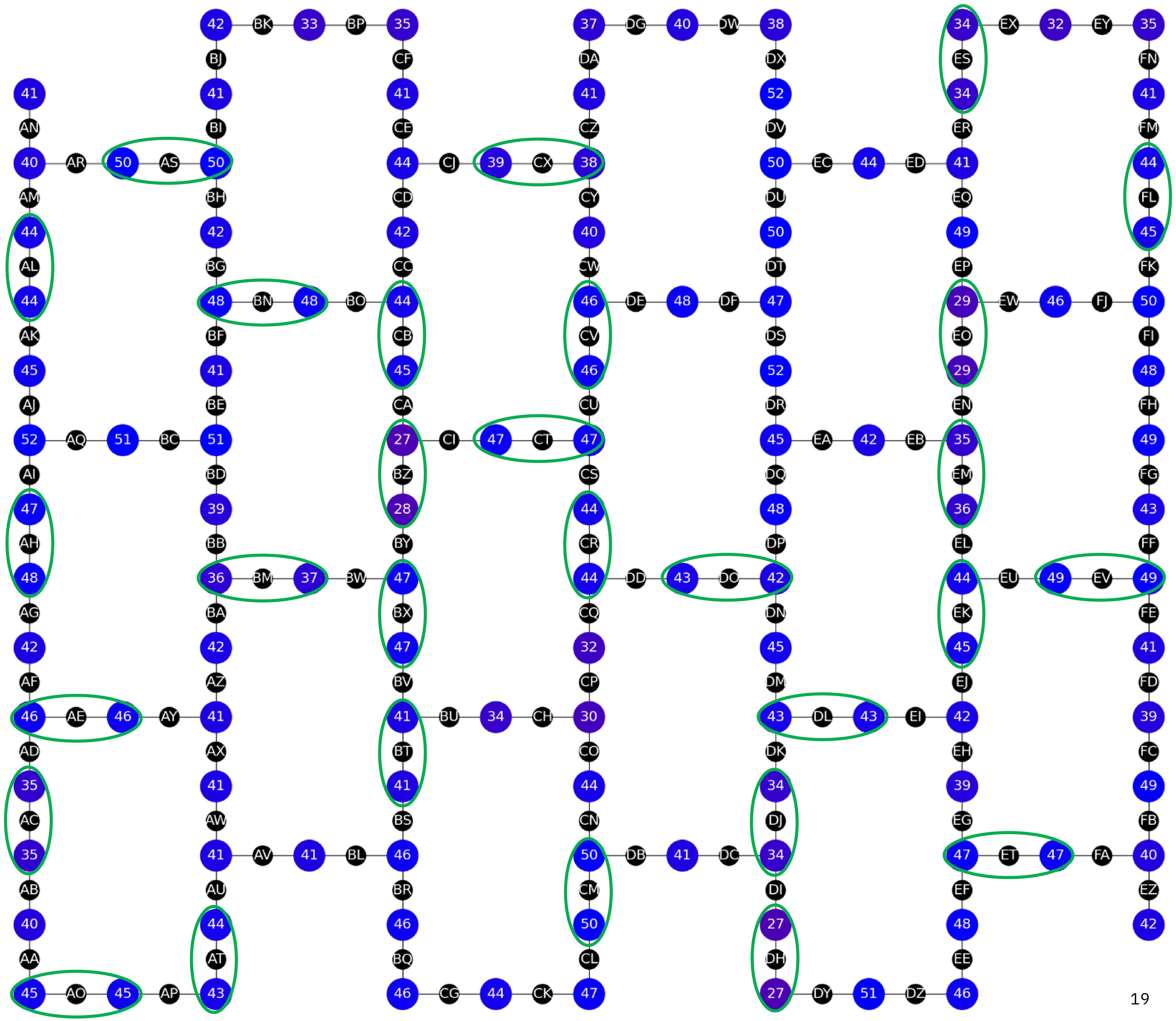
Puzzles from `ibm_kyiv`

- Example of a round 50 puzzle with cleanup



Puzzles from `ibm_kyiv`

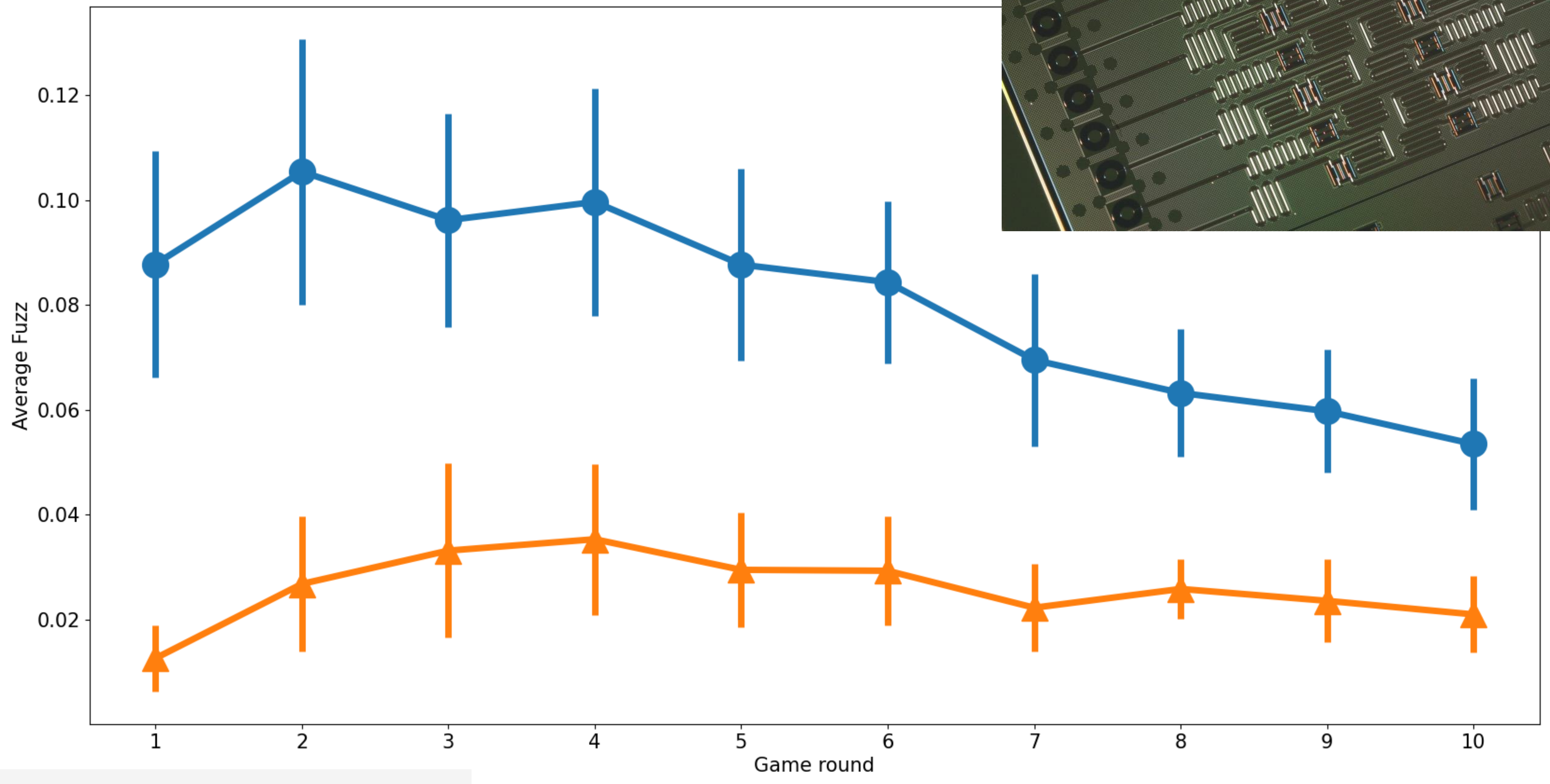
- Example of a round 50 puzzle with cleanup

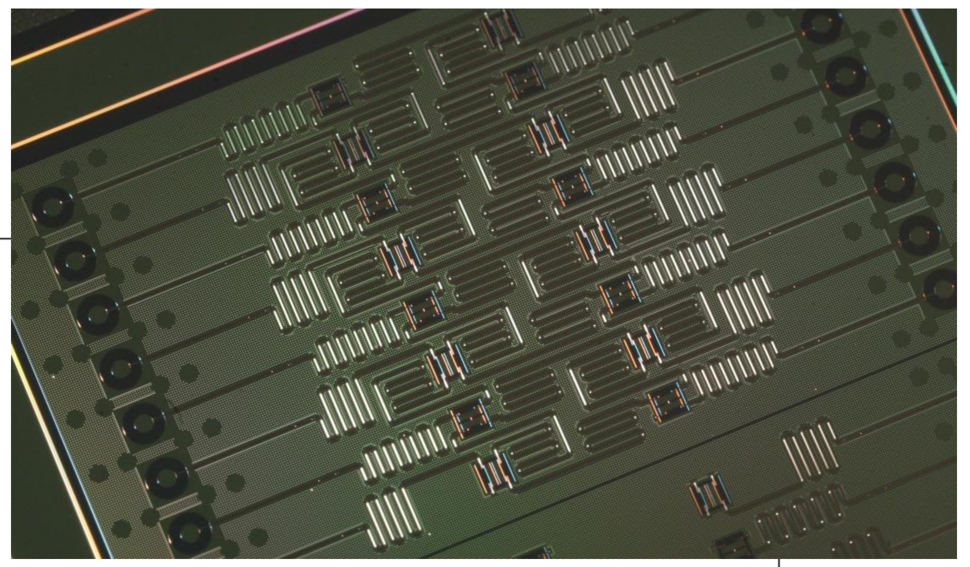


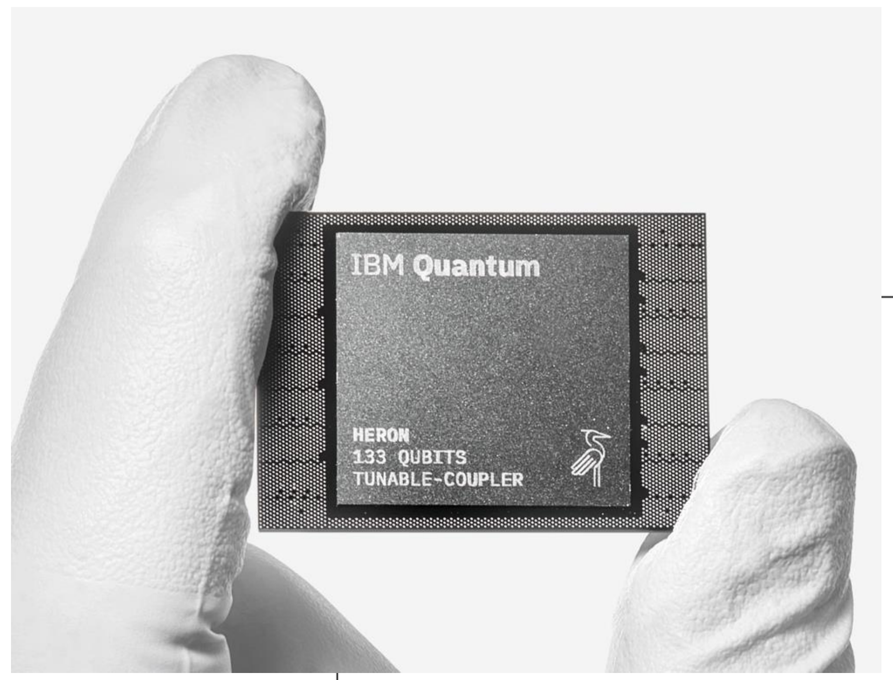
Conclusions

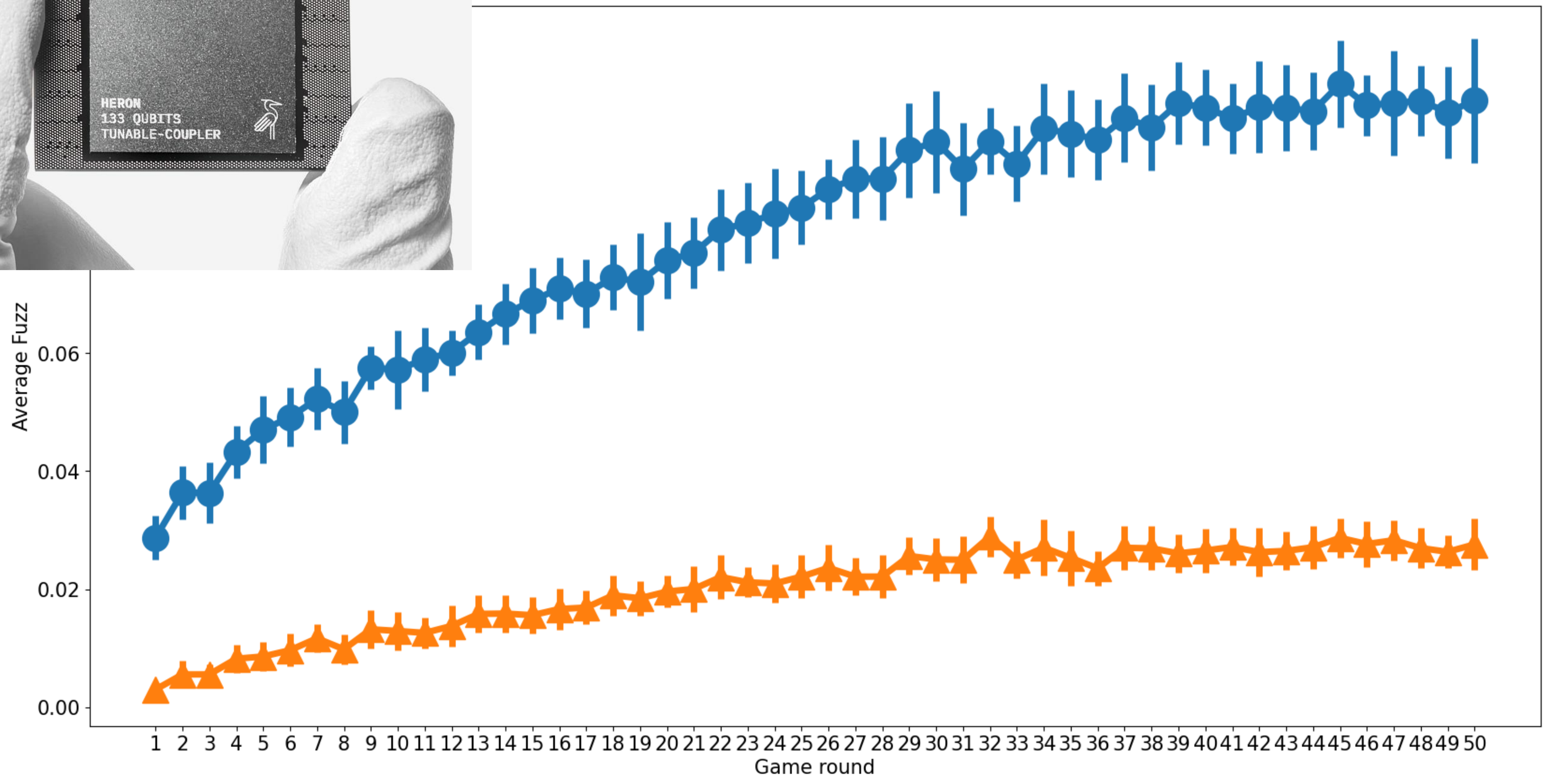
- We ran something that barely worked in 2018

- Now it works great with over 100 qubits!





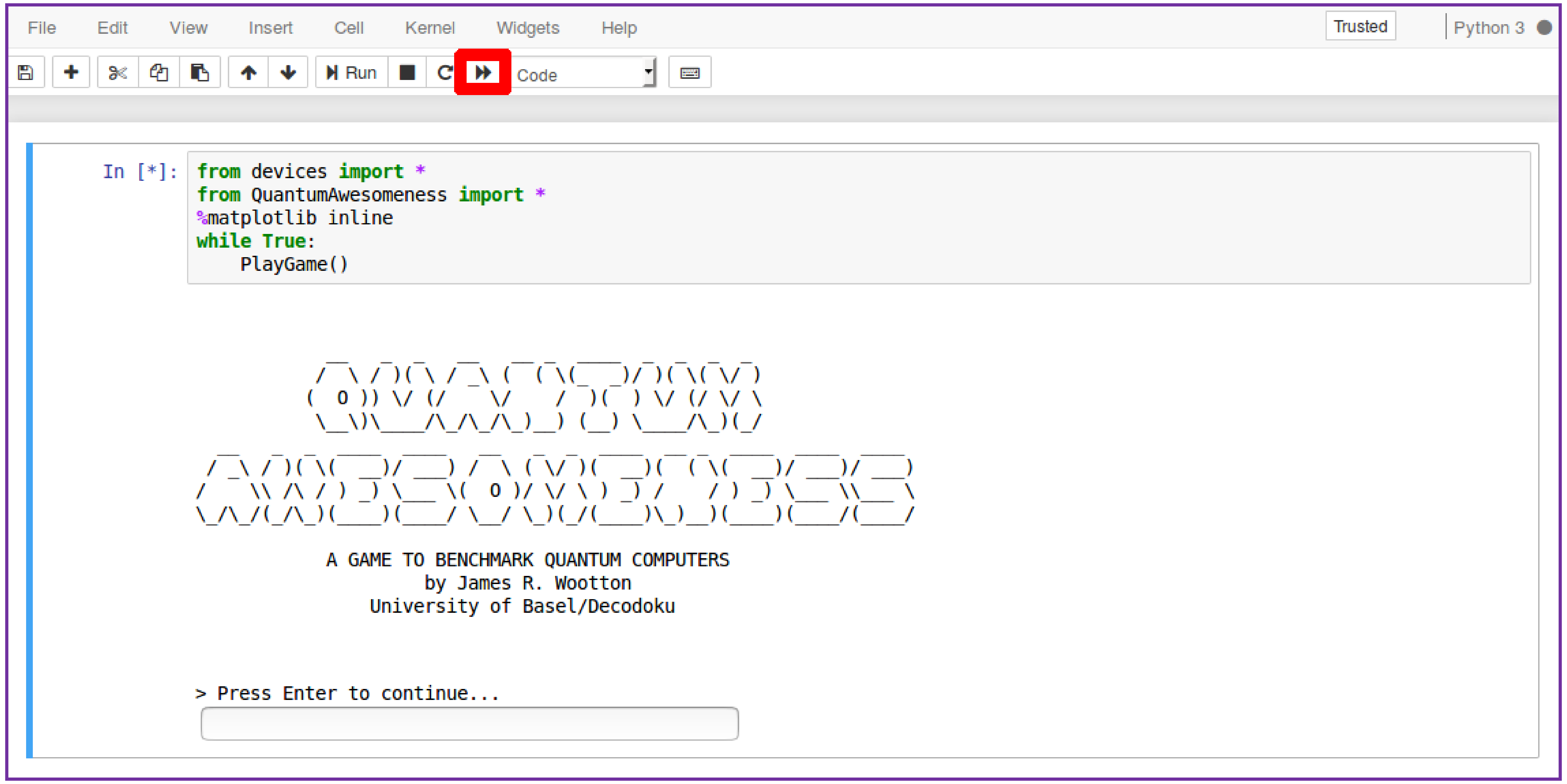




Now time to play!

- The code and data for all these games is open source
- You can try solving the puzzles yourself

- You could also use the data to make something new!
- Why not make a game with >100 qubit data this weekend?



```
In [*]: from devices import *
        from QuantumAwesomeness import *
        %matplotlib inline
        while True:
            PlayGame()
```

```
                 __   _  _   __   __ _  ____  _  _  _  _ 
                /  \ / )( \ / _\ (  ( \(_  _)/ )( \( \/ )
               (  O )) \/ (/    \/    /  )(  ) \/ (/ \/ \
                \__\)\____/\_/\_/\_)__) (__) \____/\_)(_/
  __   _  _  ____  ____   __   _  _  ____  __ _  ____  ____  ____ 
 / _\ / )( \(  __)/ ___) /  \ ( \/ )(  __)(  ( \(  __)/ ___)/ ___)
/    \\ /\ / ) _) \___ \(  O )/ \/ \ ) _) /    / ) _) \___ \\___ \
\_/\_/(_/\_)(____)(____/ \__/ \_)(_/(____)\_)__)(____)(____/(____/

               A GAME TO BENCHMARK QUANTUM COMPUTERS
                        by James R. Wootton
                    University of Basel/Decodoku


> Press Enter to continue...
```

Thanks for your attention