# Public report on Quantinuum's Quantum Circuit Tensor Network simulation challenge

Pablo Andres-Martinez

14th July 2025

## 1 Participants

Here I list the four participants of the public tier. For more information about the participants and their submissions, please visit [4].

- *pytket-mps*. The package pytket-cutensornet [1] is an open sourced library developed by Quantinuum that is used to simulate pytket circuits using NVIDIA's CuTensorNet library. The MPS algorithm is a custom implementation developed using cuTensorNet's lower level primitives "contract" and "decompose", rather than Nvidia's own MPS implementation.
- *QuantumRings*. QuantumRings SDK [5] implements the Schmidt basis using a tensorized representation. The SDK works in a CPU only mode, GPU enabled mode, and a hybrid mode.
- *qmatchatea*. Open source library for quantum circuit simulation, part of the Quantum Tea initiative [6]. Developed by the University of Padova. All results correspond to the MPS algorithm.
- *MIMIQ*. Quantum circuit emulator based on the Matrix Product State (MPS) formalism, featuring advanced circuit preconditioning and Matrix Product Operator (MPO) compression to efficiently simulate large and moderately entangled quantum systems. Find more about them at [3].

### 1.1 Hardware

Below I provide a summarised table of the compute resources that each participant used. For a more detailed description, please refer to the "README" files on each of the participants submissions at [4].

| **Participant** | **CPU** | **GPU** |
|---|---|---|
| pytket-mps | AMD EPYC 7763 | Nvidia A100 |
| Quantum Rings | — | Nvidia H100 |
| qmatchatea (default) | Intel Xeon Platinum 8358 | Nvidia A100 |
| qmatchatea (CPU) | AMD EPYC 9654 | — |
| MIMIQ | Intel Xeon w5-2545 | — |

All participants but *MIMIQ* used GPU-based simulators. Furthermore, *qmatchatea* used a CPU backend for the simulations that would run out of memory in GPU.

## 2 Validation of expectation values

The figures in this section are generated from the participant's submissions by running the script `expval_diff_heatmap.py` from [4].

We generally find good agreement between participants' submissions whenever they report a mirror fidelity $\geq 0.9$. In the case of **mvsp** circuits, we find that every participant has at least one instance where their expectation values noticeably differ from the rest. An example is shown in Fig. 1.

For these **mvsp** circuits, *MIMIQ* reports mirror fidelities of 0.90 whereas other participants are in the range $0.94 - 0.96$. Due to *MIMIQ*'s position in the ranking presented in the following section, I put special attention in validating these results, and requested higher fidelity simulations from *MIMIQ*. This brought the difference down to an acceptable margin. These higher fidelity simulations incurred longer runtime, but not sufficiently large to affect *MIMIQ*'s position in the ranking. Hence, *MIMIQ*'s submission is valid and I use their original submission in the evaluation of the results in the following section.

## 3 Results

I now present the rankings for each circuit family, according to the mirror fidelity bands specified in section 2.4 of the `Instructions.pdf` file. The figures in this section are generated from the participant's submissions by running the script `ranking.py` from [4].

In the figures, the circuits of each family are ordered according to the time it took the lead participant to run them. This aids the eye in identifying the lead participant. However, for other participants, the circuits that are easier to simulate may be different, causing the zig-zagging observed in the figures.[1]

I summarise my interpretation of the results for each circuit family separately. You may find a description of each of these circuit families in the

[1]Bar plots would avoid this caveat but, given the large number of data points, I find this visualisation provides a better summary of the results at a glance.

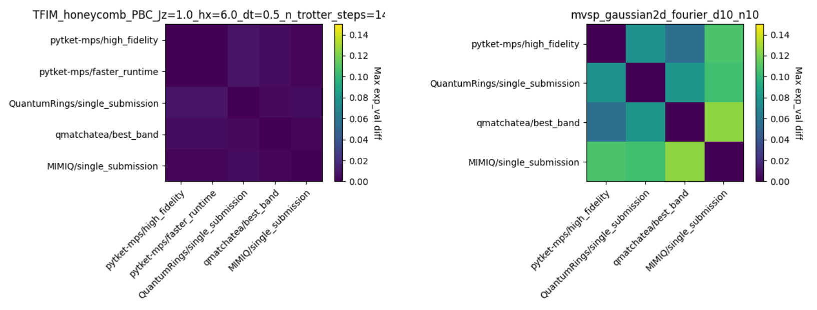

Figure 1: For each pair of participants, we indicate the largest absolute difference between expectation values across the requested observables for a given circuit. Cases where a participant had noticeably different results were investigated further, as discussed in the text.

`Instructions.pdf` at [4]. You may also find information on the number of qubits and gates for each of this circuits in the `metadata.csv` file at the same location.

**condensed_matter.** See Fig. 2. Roughly half of the circuits could not be simulated by any participant. For mirror fidelity $\geq 0.9$, *MIMIQ* leads the ranking, except for two of the harder circuits, where *qmatchatea* leads: in one case by a small runtime difference, in the other case due to being the only participant that managed to simulate the circuit. *MIMIQ* is dramatically faster in the easier circuits of this family. For mirror fidelity $\geq 0.6$ we see a similar behaviour, this time *pytket-mps* leading in one data point.

**chemistry_uccsd.** See Fig. 3. These circuits were comprised of many Pauli exponentials on multiple qubits and required special treatment to be simulated efficiently. All participants submitted mirror fidelities above $\geq 0.9$ exclusively, so both plots are identical. *pytket-mps* leads in all but four circuits: in two of them, *qmatchatea* runs faster, and in the other two, *pytket-mps* failed to submit results. *QuantumRings* is faster than *qmatchatea* on two of the harder circuits.

**mvsp.** See Fig. 4. *MIMIQ* leads across the board, with a dramatic runtime difference. *qmatchatea* comes second for most of the other circuits, and *QuantumRings* is the second for the remaining, harder, circuits. The plots for mirror fidelities $\geq 0.9$ and $\geq 0.6$ are qualitatively the same, with improved performance from *qmatchatea*.

**qmci.** See Fig. 5. I did not expect anyone would be able to simulate these circuits. *MIMIQ* is the only participant that managed to do so. The two plots look the same because *MIMIQ* provided only one submission.

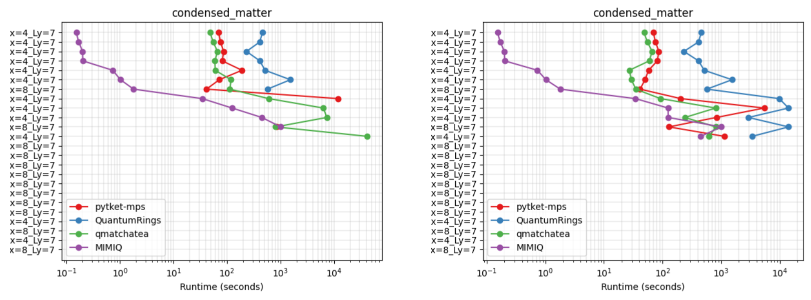

Figure 2: Results for the **condensed_matter** circuit family. On the left, only submissions that achieved mirror fidelity of $\geq 0.9$ are included. On the right, submissions with fidelity $\geq 0.6$ are included.

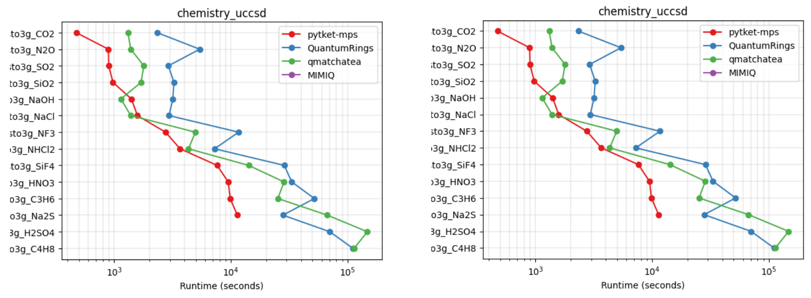

Figure 3: Results for the **chemistry_uccsd** circuit family. On the left, only submissions that achieved mirror fidelity of $\geq 0.9$ are included. On the right, submissions with fidelity $\geq 0.6$ are included.

**qec_non_ft.** See Fig. 6. These circuits are Clifford and, consequently, can be trivially simulated by stabiliser methods. These family of circuits was included in this challenge in the hopes of seeing some stabiliser-tensor network hybrid showcased by some of the participants. This was not the case, and the results show the poor scaling of tensor networks on predominantly Clifford circuits. *MIMIQ* leads in this family in all circuits that participants managed to simulate. Both plots are identical.

## 4 Conclusions

For this circuit suite, *MIMIQ* is the leader overall. Nevertheless, it is important to realise that for the harder circuits, the rest of the participants would occasionally have an edge over *MIMIQ*. I believe this reflects the fact that *MIMIQ* is the only participant running exclusively on CPUs and that, at certain point, the compute power of GPUs is necessary to achieve the highest performance.

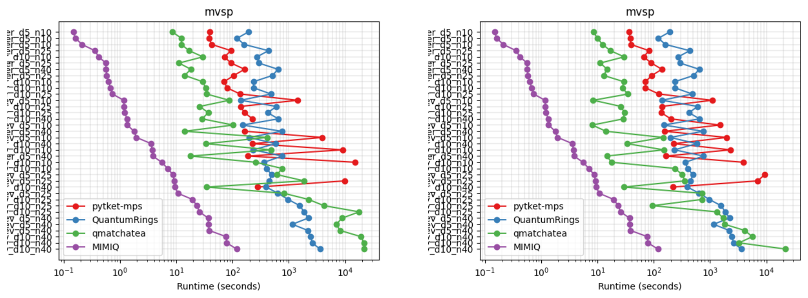

Figure 4: Results for the **mvsp** circuit family. On the left, only submissions that achieved mirror fidelity of $\geq 0.9$ are included. On the right, submissions with fidelity $\geq 0.6$ are included.

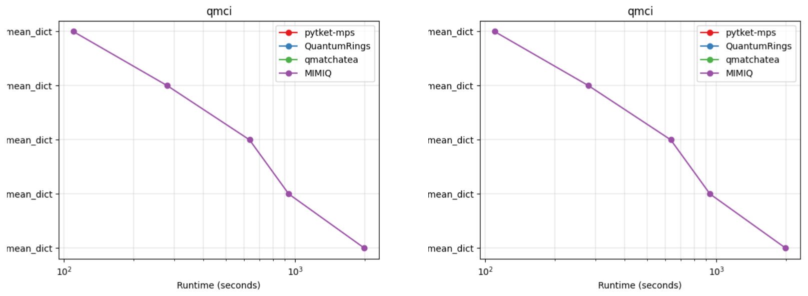

Figure 5: Results for the **qmci** circuit family. On the left, only submissions that achieved mirror fidelity of $\geq 0.9$ are included. On the right, submissions with fidelity $\geq 0.6$ are included.

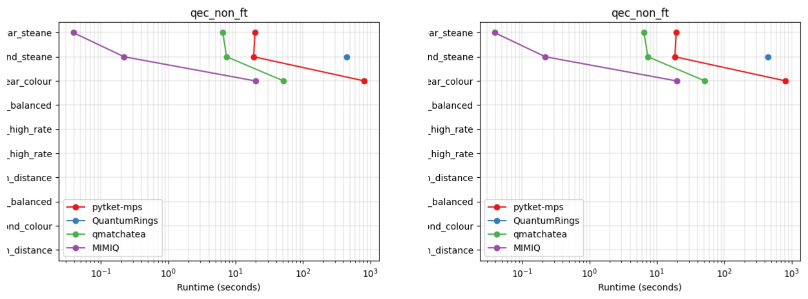

Figure 6: Results for the **qec_non_ft** circuit family. On the left, only submissions that achieved mirror fidelity of $\geq 0.9$ are included. On the right, submissions with fidelity $\geq 0.6$ are included.

More thoughts on this appear in section 4.1

*qmatchatea* is a consistent alternative that performs well on all families (except **qmci**). *pytket-mps* is comparable to *qmatchatea* for the most part, but scales worse with circuit hardness and is generally behind except in the case of the **chemistry_uccsd** family. *QuantumCircuits* is only competitive on the hardest circuits, leading sporadically.

## 4.1 GPU vs CPU crossover

We have observed crossover points between CPU and GPU simulation before. This is a commonly the case in statevector simulation [2] and we have observed it in tensor network simulation as well. I did not expect it was possible to push simulation on CPUs to be competitive at simulating circuits as hard as the ones included in this circuit suite (particularly the **qmci** and **condensed_matter** families). It is reasonable to consider that increasing the required level of fidelity may lead to simulations where GPU performs better than CPU. However, it is important to notice from Fig. 2 that the regime where GPUs are more performant than CPUs may be narrow, as we reach the point where the simulation requires more RAM than GPUs have.

## 4.2 Simulation accuracy

An immediate follow up question after evaluating these results is: "can we simulate these circuits with higher fidelity?". For the most part, the answer is "yes", but with rapidly increasing overheads. I would like to point out that a mirror fidelity of $\geq 0.9$ is quite high precision, considering many of these circuits have tens of thousands of two-qubit gates. Still, this fidelity is not high enough for some practical applications, as evidenced by the large variance in expectation values observed in the **mvsp** family. Further iterations of this challenge would benefit from requesting higher fidelities from participants.

## 4.3 Further contributions

I welcome participants (either the ones listed here or new ones) to create PRs on the public repository [4] to include new submissions in order to maintain the results of the challenge up to date with progress in the field. Please contact `pablo.andresmartinez@quantinuum.com` if you wish to contribute.

In the case of new contributions, a separate branch will be created for each version, indicating the date of the most recent submission from each of the participants.

## 4.4 To all participants: thank you!

At Quantinuum, we are aware that participating in this challenge meant a considerable investment in compute resources and employee resources. We sincerely appreciate the effort you have put into your submissions, which have all been of excellent quality.

We believe that field of tensor network simulation would benefit from more benchmarking initiatives, and we are grateful for your enthusiasm to contribute towards this.

## References

[1] *Documentation for pytket-cutensornet.* URL: `https://docs.quantinuum.com/tket/extensions/pytket-cutensornet/`.

[2] Amit Jamadagni Gangapuram, Andreas Läuchli, and Cornelius Hempel. "Benchmarking quantum computer simulation software packages: State vector simulators". In: *SciPost Physics Core* 7.4 (Nov. 2024). ISSN: 2666-9366. DOI: `10.21468/scipostphyscore.7.4.075`. URL: `http://dx.doi.org/10.21468/SciPostPhysCore.7.4.075`.

[3] *MIMIQ main page.* URL: `https://qperfect.io/index.php/mimiq/`.

[4] *Public repository with the participant's submissions.* URL: `https://github.com/CQCL/public_tn_sim_challenge_results`.

[5] *Quantum Rings page.* URL: `https://www.quantumrings.com`.

[6] *Quantum Tea page.* URL: `https://www.quantumtea.it/`.