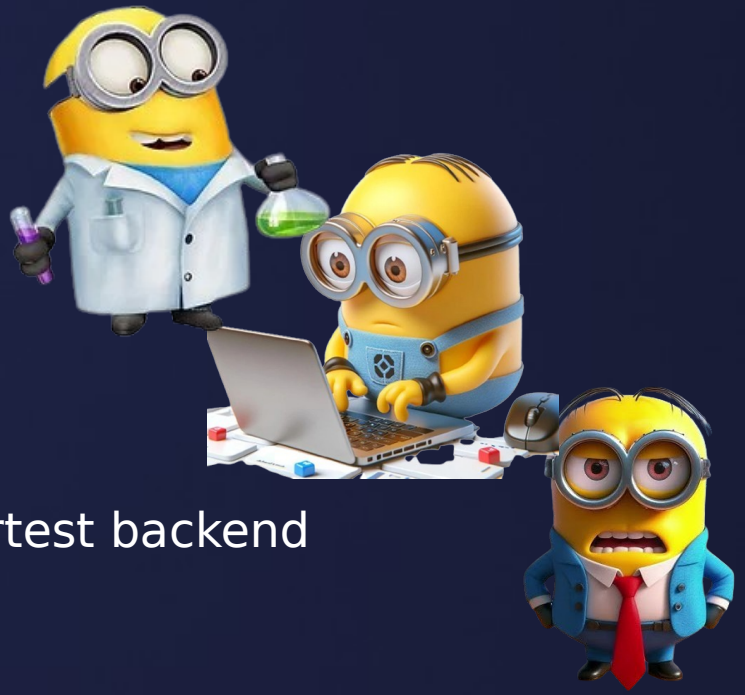
Quantum Emulator Benchmarking Made Easy

Authors:
Anna Leonteva,
Maxime Outteryck,
Guido Masella

Why Benchmark Quantum Emulators?

Depending on who you are...



Researcher: to understand limits and accuracy

Developer: to compare methods fairly

Engineer: to optimize configurations

Application product owner: to choose the smartest backend

But ultimately:

Benchmarking shows what actually works and when.

What makes benchmarking difficult?

- Too many emulator-specific settings
- Different programming languages & APIs
- Not obvious how to ensure a fair comparison
- Emulator performance depends heavily on internal parameters
- Possible versioning conflicts (e.g., incompatible Python package versions)
- Timeouts and crashes complicate evaluation
- Difficult to reproduce experiment across environments

What FENIQS is ?

A quantum-emulator benchmarking framework that is

- Open-source
- User-centric
- Problem-agnostic
- Unified API for 12 emulators
- Plugin-based, modular, extensible

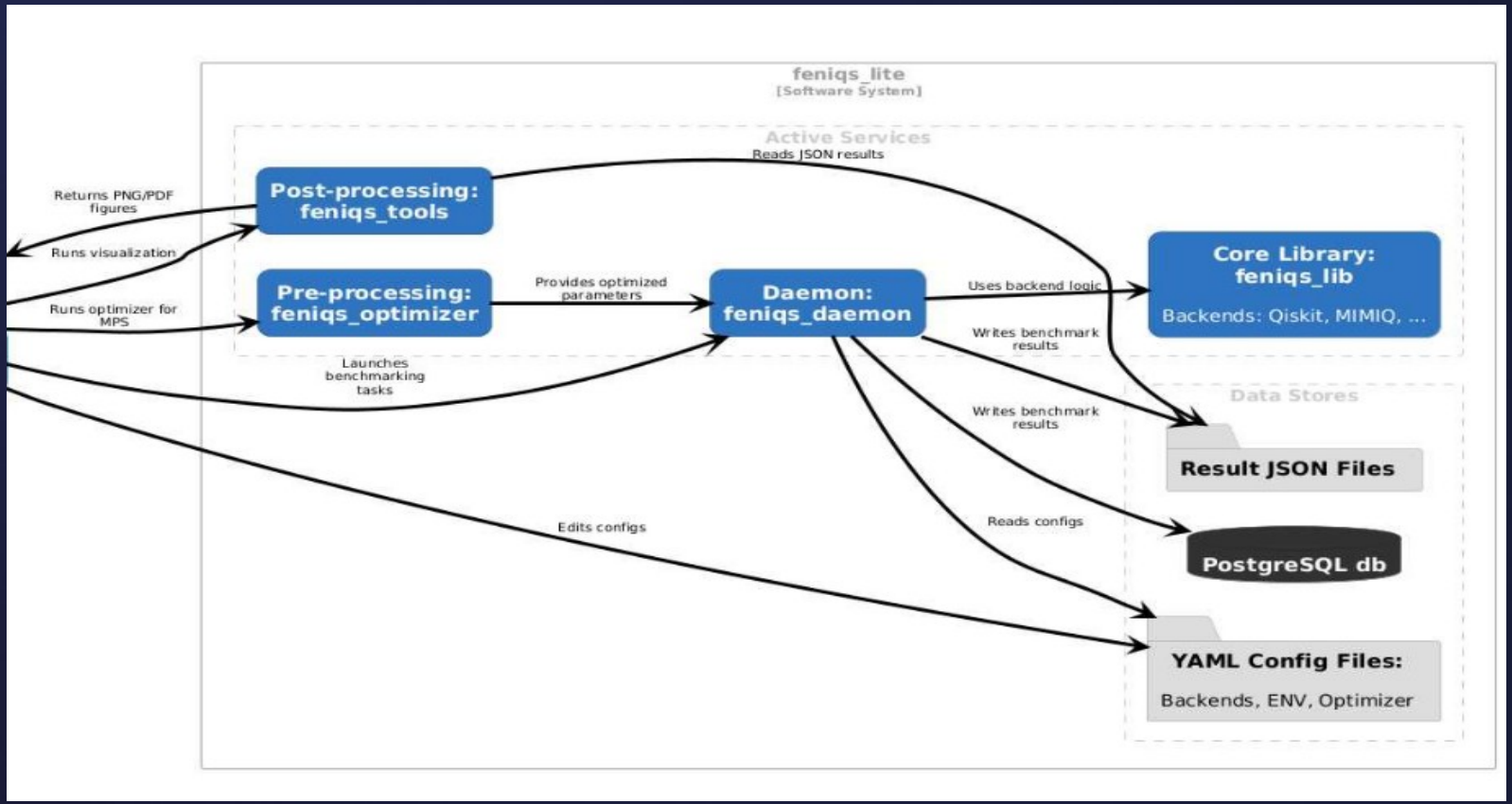
FROM

TO

What FENIQS does ?

- Automates the entire benchmarking workflow
  — fast setup, no coding required
- Runs OpenQASM circuits across multiple emulators
  — flexible scenarios, unified interface
- Enforces timeouts
  — reliable, controlled execution
- Stores clean, structured results
  — PostgreSQL or JSON
- Provides ready-to-use visualization scripts
- Makes benchmarking easier and users happier

Architecture Overview



Supported Backends

Summary:

12 frameworks
6 emulators
CPU/GPU impl.
2 API Lang

N	Framework	Method	API Lang.	Open Source	CPU	GPU
1	Cirq	SV	Python	yes	yes	yes
2	Qiskit	SV / MPS	Python	yes	yes	yes
3	MIMIQ	SV / MPS	Julia	no	yes	no*
4	Pennylane	SV	Python	yes	yes	yes
5	Qulacs	SV	Python	yes	yes	yes
6	Yao	SV	Julia	yes	yes	yes
7	ProjectQ	SV	Python	yes	yes	no
8	QMatchaTea	MPS	Python	yes	yes	yes
9	MQT-DDS	Decision Diagrams	Python	yes	yes	no
10	Pyqrack	Factorized Ket	Python	yes	yes	no
11	Qrisp	Sprase Matrix	Python	yes	yes	no
12	Quimb	MPS / NT	Python	yes	yes	no

Metrics

1. Run time:
   total wall-clock time required to execute a quantum circuit

2. Mirror-Circuit Fidelity (for MPS):

$$F_{\text{mirror}} = \left|\langle\psi_0|U^\dagger U|\psi_0\rangle\right|^2 = p_{\psi_0},$$

3. Scalability:

$$\mathcal{T}_{\mathcal{E}}(n) = \text{Run time of emulator } \mathcal{E} \text{ for } n \text{ qubits, subject to } \mathcal{F}_{\mathcal{E}}(n) \geq 0.99,$$

Hyperparameter Optimization

- Only for MPS
- Noise-aware parameter search
- Single-objective optimization:

  CMA-ES
- Multi-objective optimization:

  NSGA-II / MOEA-D

Circuit	MIMIQ-MPS	Qiskit-MPS
qft	[4/4/1e-5], [vmpoa/T/F]	[4/1e-10/AM],[1/T]
qftentangled	[4/4/1e-5],[vmpoa/T/F]	[4/1e-10/AM],[1/F]
ghz	[4/4/1e-5], [vmpoa/T/F]	[4/1e-10/P],[0/T]
wstate	[4/4/1e-5], [vmpoa/T/F]	[4/1e-10/P],[0/T]
qpeexact	[4/4/1e-5],[vmpoa/T/F]	[4/1e-10/P],[1/T]
qpeinexact	[4/4/1e-5],[vmpoa/T/F]	[4/1e-10/P],[1/T]
qwalk	[32/8/1e-5],[vmpoa/T/F]	[32/1e-5/P],[0/F]
ae	[64/4/1e-3],[vmpoa/T/F]	[32/1e-5/P],[1/F]
realamp	[32/4/1e-5],[vmpoa/T/F]	[1024/1e-6/P],[1/T]
su2rand	[32/4/1e-5],[vmpoa/T/F]	[1024/1e-6/P],[1/T]
qnn	[384/4/1e-5],[dmpo/T/F]	[256/1e-5/P],[1/T]
graphstate	[256/16/1e-5],[vmpoa/T/T]	[2048/1e-10/P],[1/T]
random	[512/8/1e-5],[vmpoa/T/T]	[2048/1e-5/P],[1/T]

MIMIQ-MPS: [bond_dimension / entdim / scut], [meth / fuse / perm]

Qiskit-MPS: [matrix_product_state_max_bond_dimension / matrix_product_state_truncation_threshold / mps_sample_measure_algorithm],

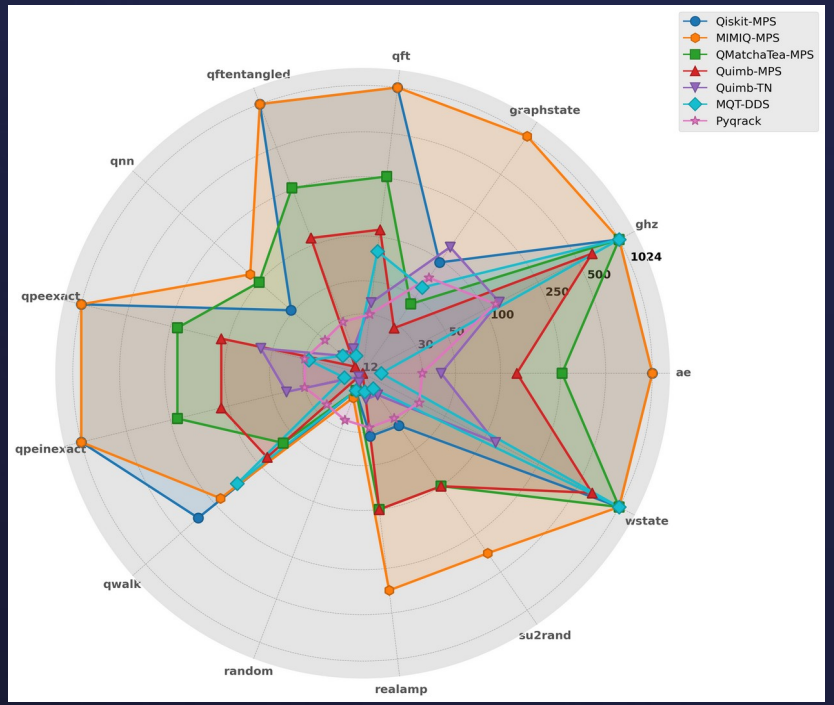
Main feature of FENIQS

It saves your time for

task	manual (min)	feniqs_lite (min)	Saved (%)
Emulator installation	1 – 15 (7)	0	100%
Test-case setup	10 – 70 (40)	0	100%
Data export + visual.	10 – 70 (40)	0	100%
Monitoring	*	0	100 %
Total (avg.)	26 – 195 (110.5)	0	100 %

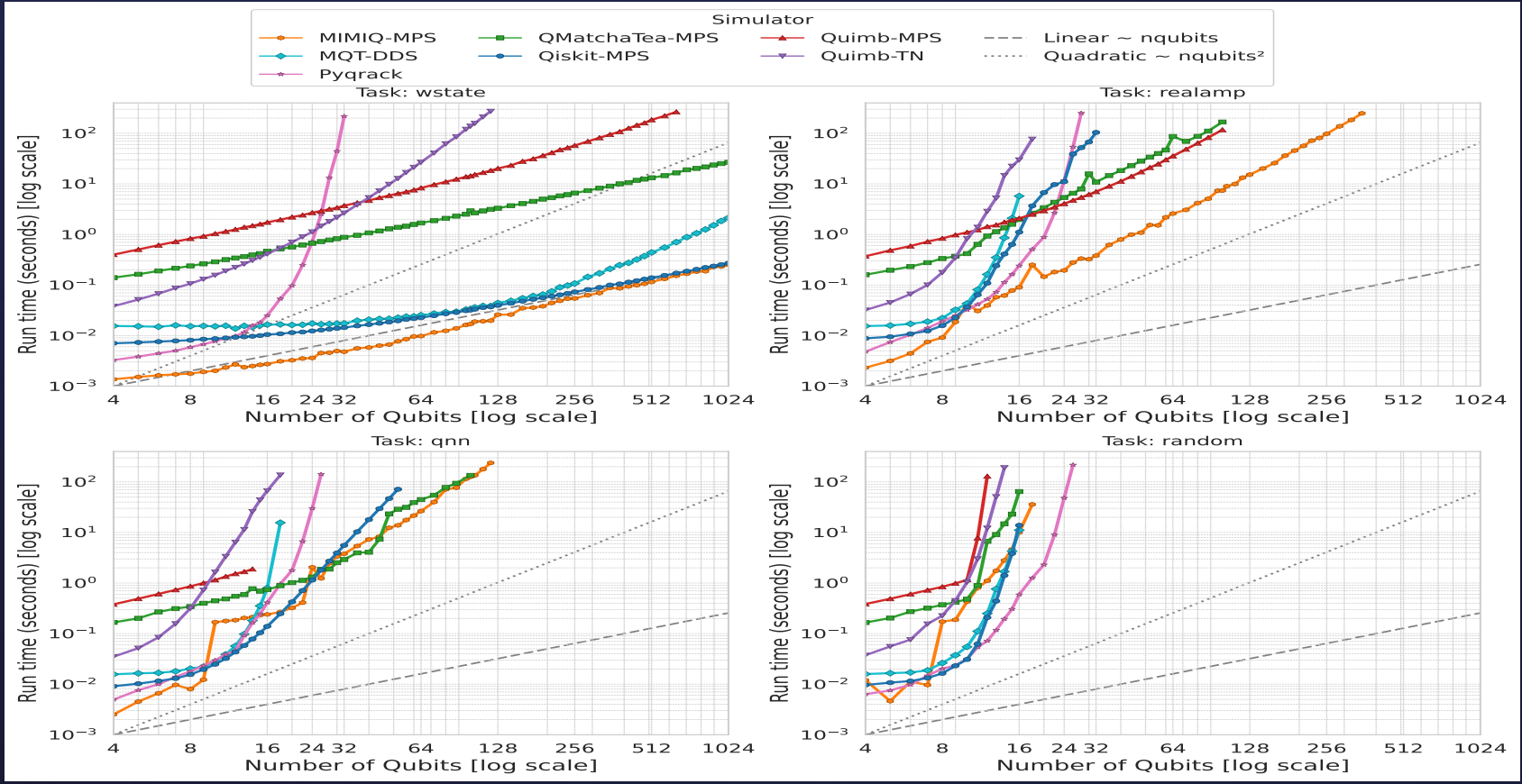
Proven at scale

Comparative Benchmarking of Utility-Scale Quantum Emulators
Anna Leonteva, Guido Masella, Maxime Outteryck, Asier Piñeiro Orioli, Shannon Whitlock
ACM Transactions on Quantum Computing, November 2025
DOI: 10.1145/3776567

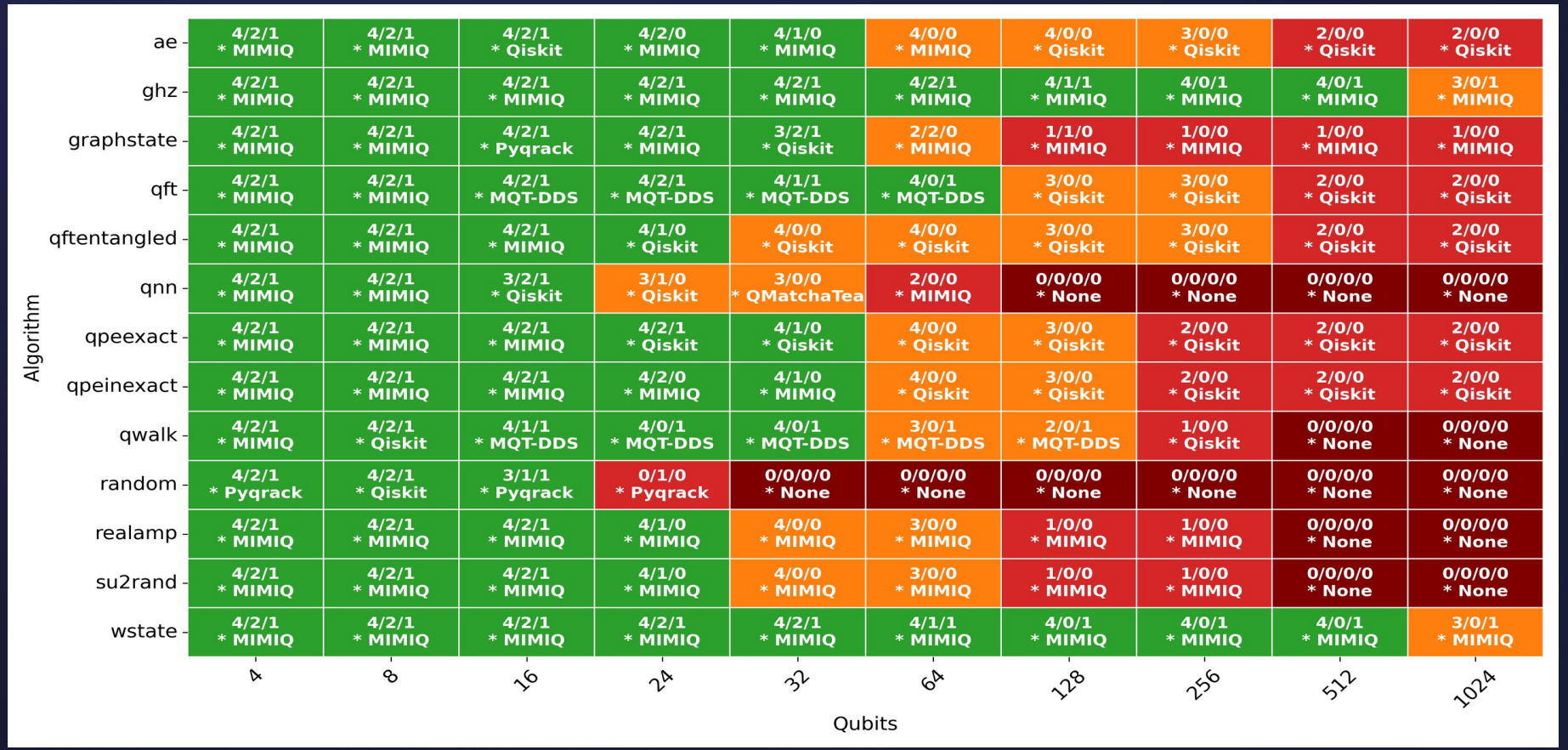


- Benchmarked 7 state-of-the-art emulators
- 13 scalable circuits
- Up to 1024 qubits
- Clear performance maps
- Practical insight into simulation limits

Run time Results



Measured Circuit Complexity



Gamification

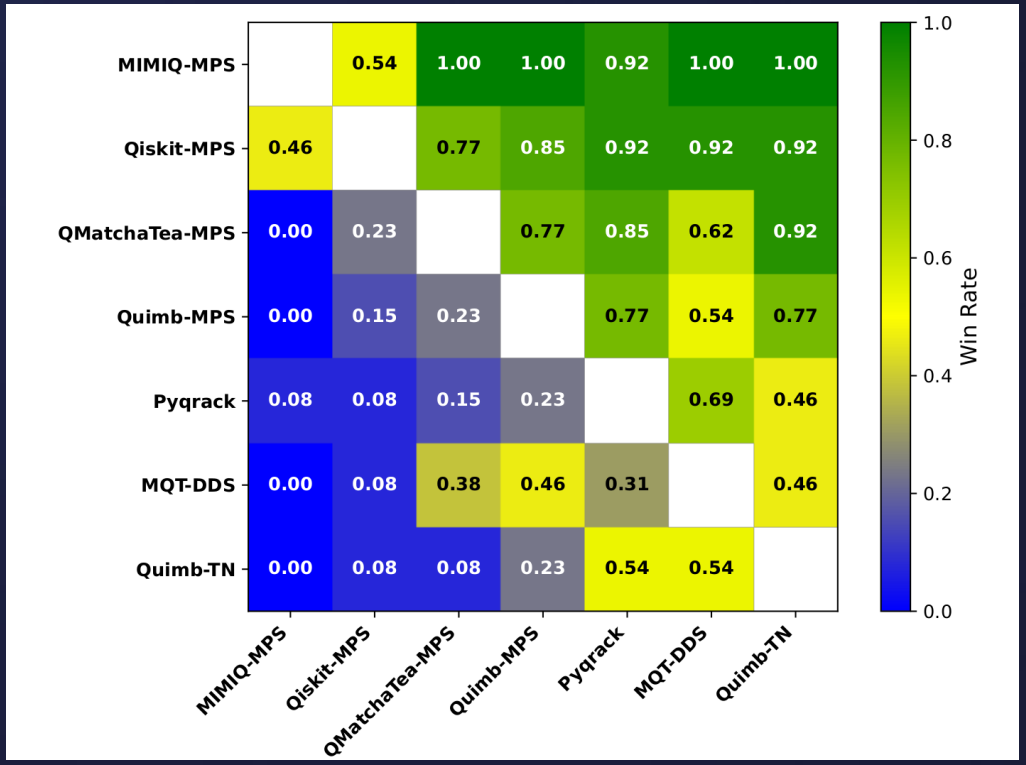
ELO-based Ranking

$$R'_A = R_A + K(S_A - E_A),$$

K =32, S_A = 1 if win else 0

$$E_A = \frac{1}{1 + 10^{(R_B - R_A)/400}}.$$

Win rates between emulator pairs



Follow us

QUEST-IS'25
QUANTUM ENGINEERING SCIENCES & TECHNOLOGIES FOR INDUSTRY & SERVICES

https://github.com/qperfect-io/feniqs_lite

Questions ?