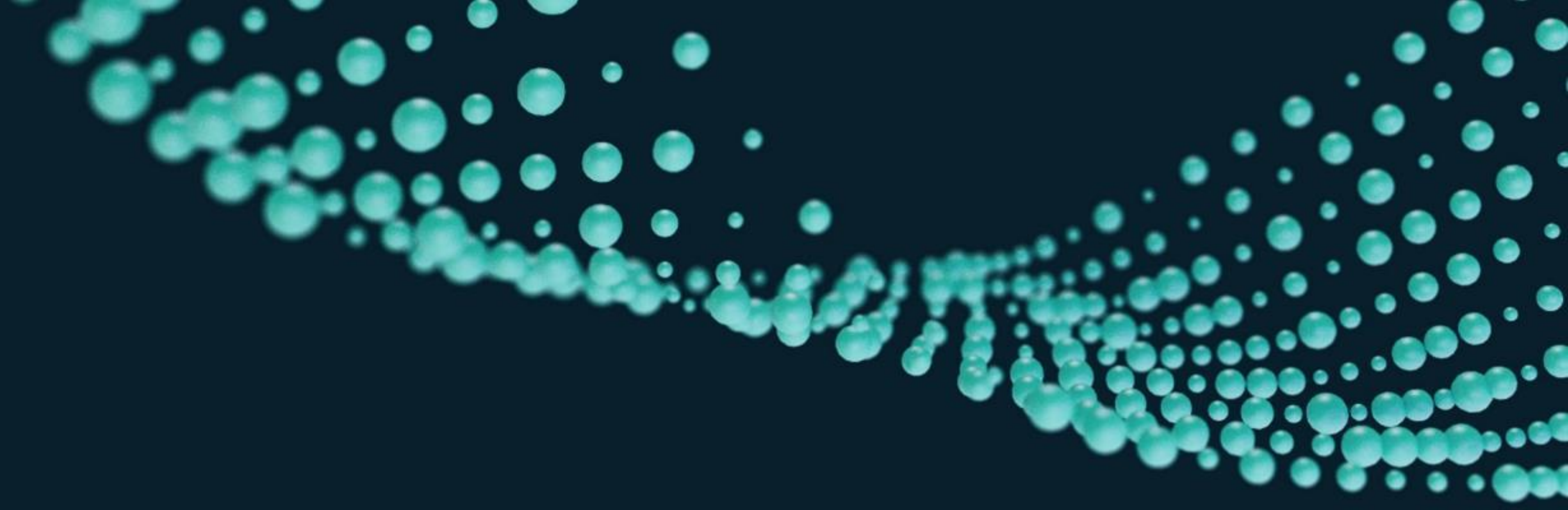


Measuring Performance of Fault Tolerant Quantum Computation

Wim van Dam
Principal Researcher
Advanced Quantum Development
Microsoft

FAULT TOLERANCE

Achieving reliable quantum computing

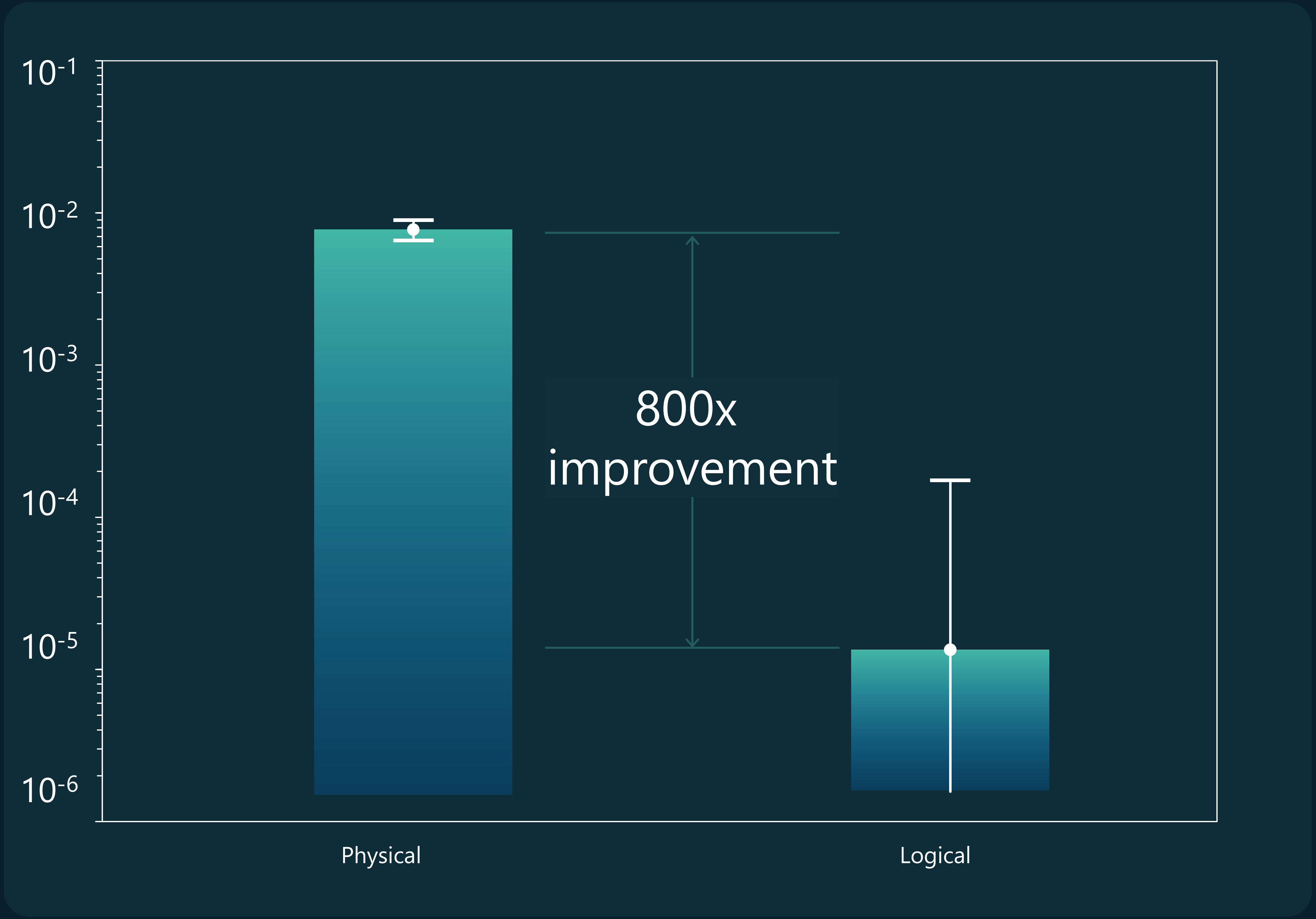
Recently, Microsoft & Quantinuum:

arXiv:2404.02280: "Demonstration of logical qubits and repeated error correction with better-than-physical error rates", April 2024

On Quantinuum's H2 machine, entangled logical qubits exhibit a circuit error rate of 10^{-5} versus a physical circuit error rate of $8x10^{-3}$.

For the first time, successfully demonstrated multiple rounds of active syndrome extractions, a critical component of reliable quantum computing

Required ~350 physical two-qubit operations and over 75 physical timesteps



RESILIENT QUANTUM COMPUTING

Quantum Computing Implementation Levels

Level 1 | **Foundational**

Physical qubits

Quantum advantage:

~60 physical qubits, contrived problems

Level 2 | **Resilient**

Logical qubits

Scientific advantage:

100 logical qubits, 10^{-8} error rate

Level 3 | **Scale**

Quantum supercomputer

Commercial advantage:

1000 logical qubits, 10^{-12} error rate

Reliable Quantum Operations Per Second (rQOPS)

$$R = Q \times f, \ \text{at } \varepsilon_L \text{ error rate}$$

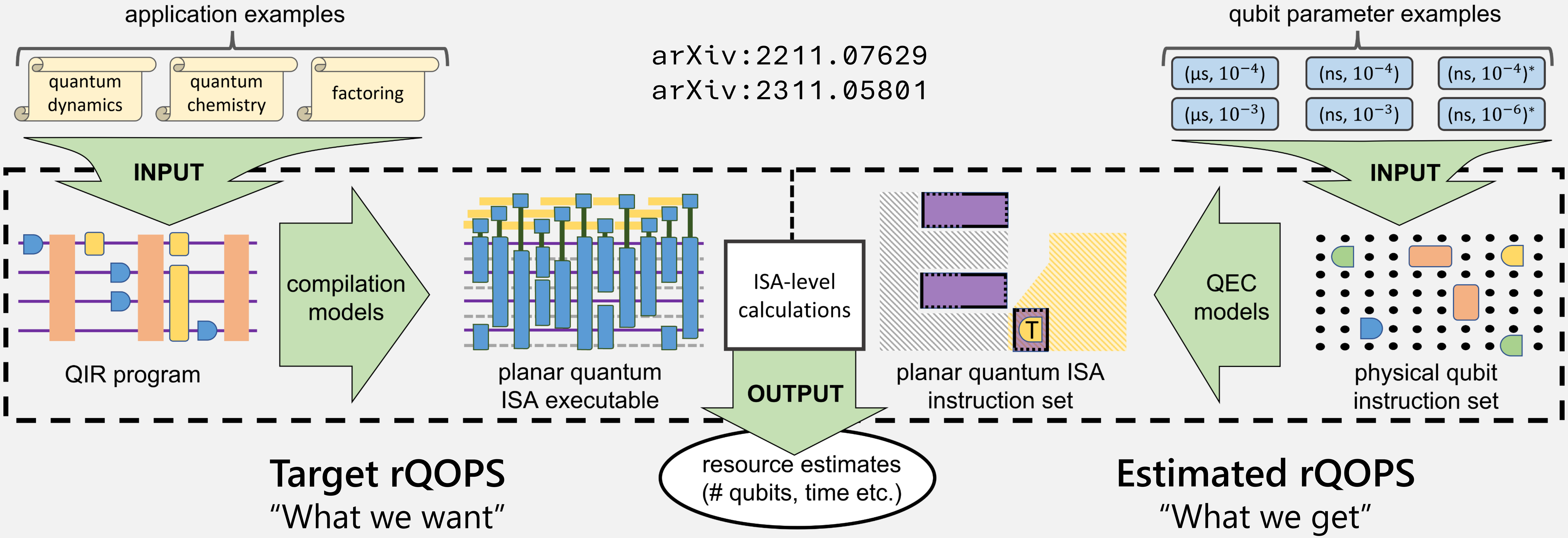
Q: number of reliable logical qubits

f: logical clock speed (in hertz)

ε_L: logical error rate

Azure Quantum Resource Estimator

arXiv:2211.07629
arXiv:2311.05801



Target rQOPS
"What we want"

Estimated rQOPS
"What we get"

Based on:
- Size of logical circuit we want to execute
- Target error of outcome
- Target run-time

Based on:
- Hardware and systems architecture parameters
- QEC scheme + distance used

Four Sample Applications

- Ising model
- Hubbard model
- Heisenberg model
- Quantum chemistry

SAMPLE 1/4

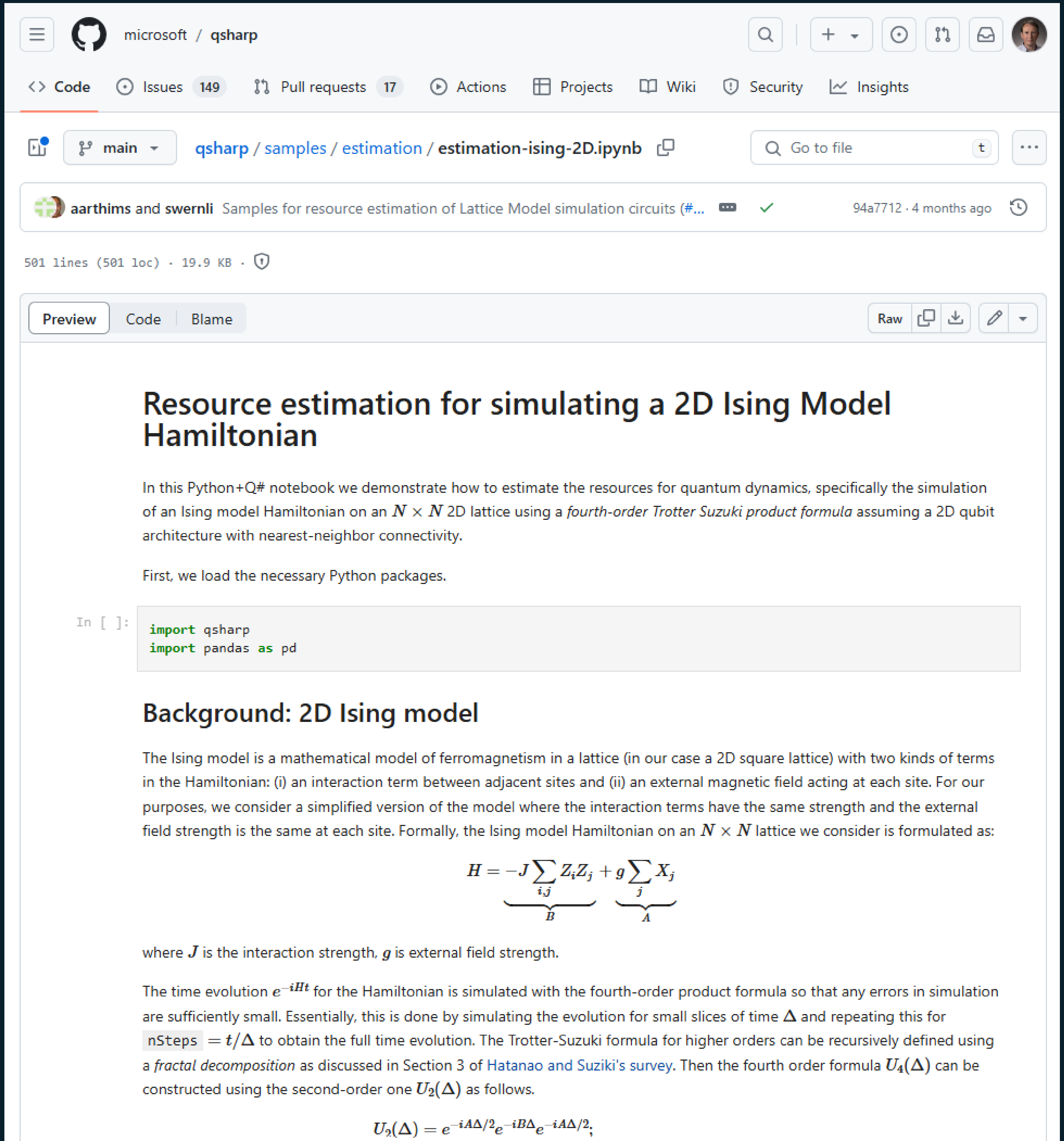
Ising Model Dynamics

- https://aka.ms/AQRE/2DIsing

- Logical resource estimates for $N \times N$ Ising model

	10×10	20×20	30×30
qubits	230	858	1886
depth	9.0×10^4	5.8×10^5	1.9×10^6
error	1.6×10^{-10}	6.7×10^{-12}	9.5×10^{-13}
target rQOPS	119	2881	20399

- Target run-time: 2 days



Samples for resource estimation of Lattice Model simulation circuits (#...

501 lines (501 loc) · 19.9 KB ·

Resource estimation for simulating a 2D Ising Model Hamiltonian

In this Python+Q# notebook we demonstrate how to estimate the resources for quantum dynamics, specifically the simulation of an Ising model Hamiltonian on an $N \times N$ 2D lattice using a *fourth-order Trotter Suzuki product formula* assuming a 2D qubit architecture with nearest-neighbor connectivity.

First, we load the necessary Python packages.

```
In [ ]:
import qsharp
import pandas as pd
```

Background: 2D Ising model

The Ising model is a mathematical model of ferromagnetism in a lattice (in our case a 2D square lattice) with two kinds of terms in the Hamiltonian: (i) an interaction term between adjacent sites and (ii) an external magnetic field acting at each site. For our purposes, we consider a simplified version of the model where the interaction terms have the same strength and the external field strength is the same at each site. Formally, the Ising model Hamiltonian on an $N \times N$ lattice we consider is formulated as:

$$H = \underbrace{-J\sum_{i,j} Z_i Z_j}_{B} + \underbrace{g\sum_j X_j}_{A}$$

where J is the interaction strength, g is external field strength.

The time evolution e^{-iHt} for the Hamiltonian is simulated with the fourth-order product formula so that any errors in simulation are sufficiently small. Essentially, this is done by simulating the evolution for small slices of time Δ and repeating this for `nSteps` $= t/\Delta$ to obtain the full time evolution. The Trotter-Suzuki formula for higher orders can be recursively defined using a *fractal decomposition* as discussed in Section 3 of Hatanao and Suziki's survey. Then the fourth order formula $U_4(\Delta)$ can be constructed using the second-order one $U_2(\Delta)$ as follows.

$$U_2(\Delta) = e^{-iA\Delta/2} e^{-iB\Delta} e^{-iA\Delta/2};$$

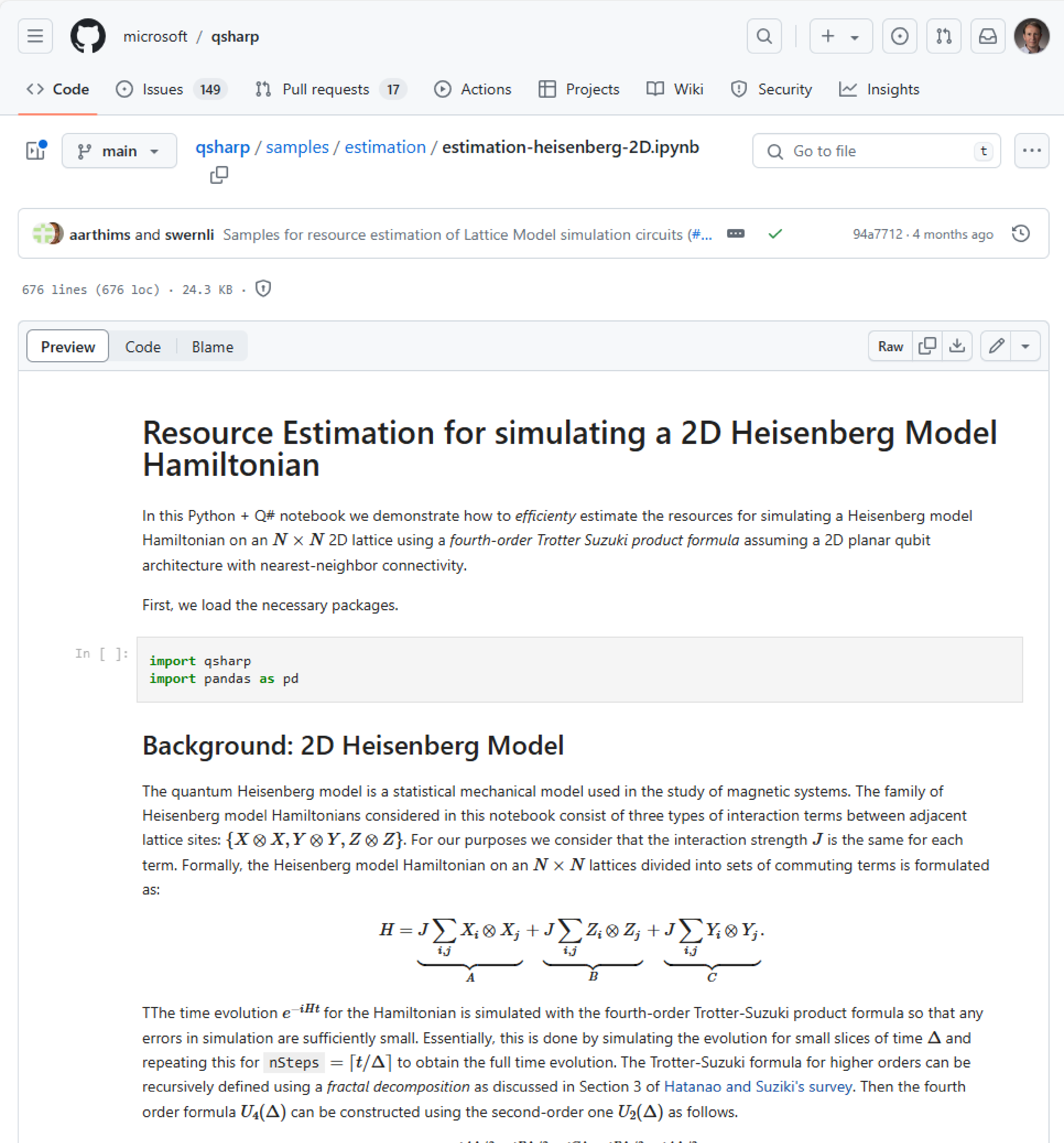
SAMPLE 2/4

Heisenberg Model

- https://aka.ms/AQRE/2DHeisenberg
- Logical resource estimates for $N \times N$ Heisenberg model

	20×20	30×30	40×40
qubits	858	1886	3315
depth	6.9×10^{8}	1.5×10^{9}	2.6×10^{9}
error	5.5×10^{-15}	1.2×10^{-15}	3.8×10^{-16}
target rQOPS	1.0×10^{6}	4.7×10^{6}	1.4×10^{7}

- Target run-time: 1 week



aarthims and swernli Samples for resource estimation of Lattice Model simulation circuits (#... 94a7712 · 4 months ago

676 lines (676 loc) · 24.3 KB ·

Resource Estimation for simulating a 2D Heisenberg Model Hamiltonian

In this Python + Q# notebook we demonstrate how to *efficienty* estimate the resources for simulating a Heisenberg model Hamiltonian on an $N \times N$ 2D lattice using a *fourth-order Trotter Suzuki product formula* assuming a 2D planar qubit architecture with nearest-neighbor connectivity.

First, we load the necessary packages.

In [ ]:
```
import qsharp
import pandas as pd
```

Background: 2D Heisenberg Model

The quantum Heisenberg model is a statistical mechanical model used in the study of magnetic systems. The family of Heisenberg model Hamiltonians considered in this notebook consist of three types of interaction terms between adjacent lattice sites: $\{X \otimes X, Y \otimes Y, Z \otimes Z\}$. For our purposes we consider that the interaction strength J is the same for each term. Formally, the Heisenberg model Hamiltonian on an $N \times N$ lattices divided into sets of commuting terms is formulated as:

$$H = J \underbrace{\sum_{i,j} X_i \otimes X_j}_{A} + J \underbrace{\sum_{i,j} Z_i \otimes Z_j}_{B} + J \underbrace{\sum_{i,j} Y_i \otimes Y_j}_{C}.$$

TThe time evolution e^{-iHt} for the Hamiltonian is simulated with the fourth-order Trotter-Suzuki product formula so that any errors in simulation are sufficiently small. Essentially, this is done by simulating the evolution for small slices of time Δ and repeating this for `nSteps` $= \lceil t/\Delta \rceil$ to obtain the full time evolution. The Trotter-Suzuki formula for higher orders can be recursively defined using a *fractal decomposition* as discussed in Section 3 of Hatanao and Suziki's survey. Then the fourth order formula $U_4(\Delta)$ can be constructed using the second-order one $U_2(\Delta)$ as follows.

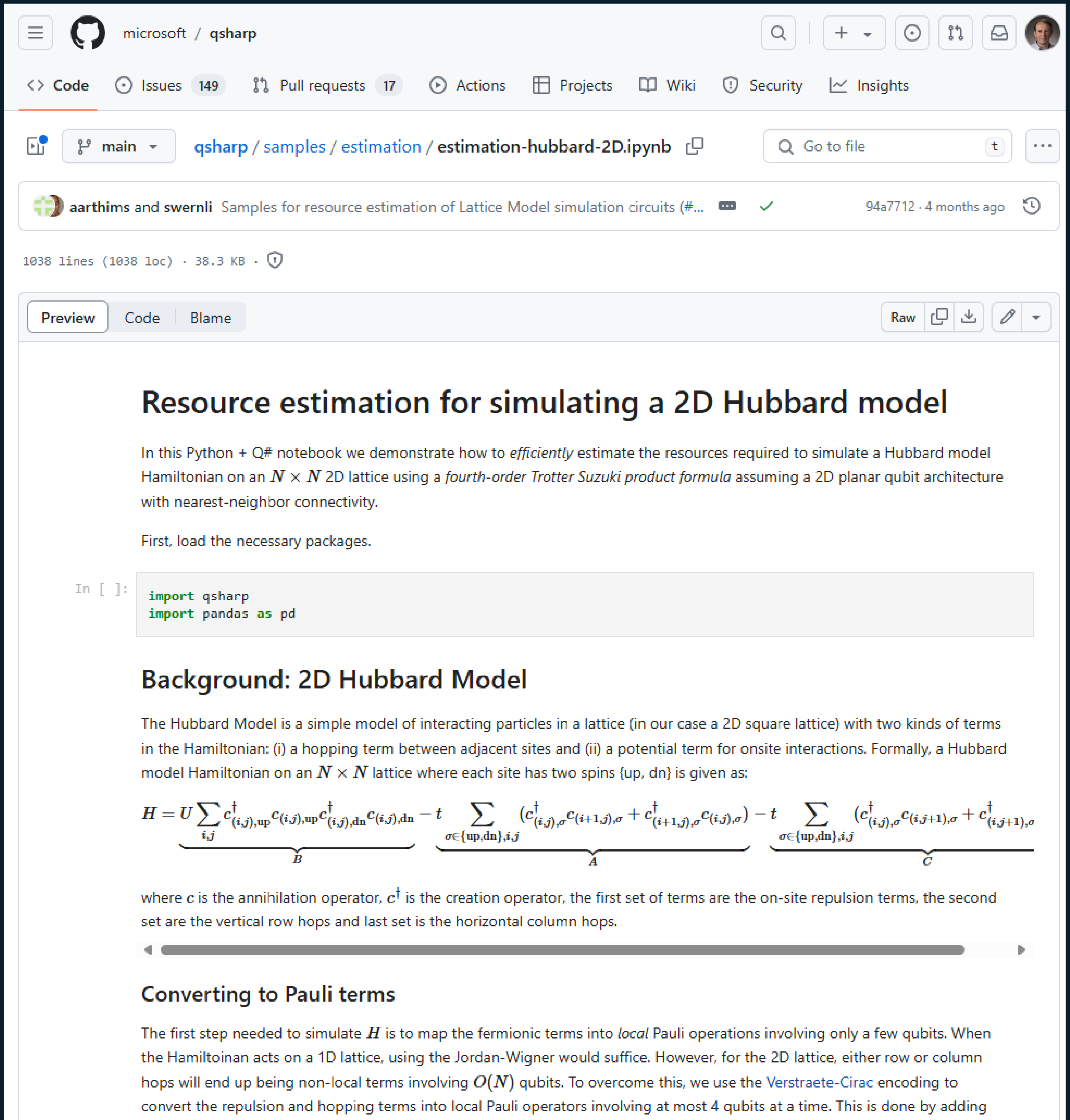
SAMPLE 3/4

Hubbard Model

- https://aka.ms/AQRE/2DHubbard
- Logical resource estimates for $N \times N$ Hubbard model

	20×20	30×30	40×40
qubits	3315	7371	13028
depth	1.2×10^{9}	2.5×10^{9}	4.5×10^{9}
error	8.5×10^{-16}	1.8×10^{-16}	5.6×10^{-17}
target rQOPS	6.5×10^{6}	3.2×10^{7}	9.8×10^{7}

- Target run-time: 1 week



aarthims and swernli Samples for resource estimation of Lattice Model simulation circuits (#... 94a7712 · 4 months ago

1038 lines (1038 loc) · 38.3 KB ·

Resource estimation for simulating a 2D Hubbard model

In this Python + Q# notebook we demonstrate how to *efficiently* estimate the resources required to simulate a Hubbard model Hamiltonian on an $N \times N$ 2D lattice using a *fourth-order Trotter Suzuki product formula* assuming a 2D planar qubit architecture with nearest-neighbor connectivity.

First, load the necessary packages.

```
In [ ]:
import qsharp
import pandas as pd
```

Background: 2D Hubbard Model

The Hubbard Model is a simple model of interacting particles in a lattice (in our case a 2D square lattice) with two kinds of terms in the Hamiltonian: (i) a hopping term between adjacent sites and (ii) a potential term for onsite interactions. Formally, a Hubbard model Hamiltonian on an $N \times N$ lattice where each site has two spins {up, dn} is given as:

$$H = \underbrace{U \sum_{i,j} c^{\dagger}_{(i,j),\mathrm{up}} c_{(i,j),\mathrm{up}} c^{\dagger}_{(i,j),\mathrm{dn}} c_{(i,j),\mathrm{dn}}}_{B} - t \underbrace{\sum_{\sigma \in \{\mathrm{up},\mathrm{dn}\},i,j} (c^{\dagger}_{(i,j),\sigma} c_{(i+1,j),\sigma} + c^{\dagger}_{(i+1,j),\sigma} c_{(i,j),\sigma})}_{A} - t \underbrace{\sum_{\sigma \in \{\mathrm{up},\mathrm{dn}\},i,j} (c^{\dagger}_{(i,j),\sigma} c_{(i,j+1),\sigma} + c^{\dagger}_{(i,j+1),\sigma}}_{C}$$

where c is the annihilation operator, $c^{\dagger}$ is the creation operator, the first set of terms are the on-site repulsion terms, the second set are the vertical row hops and last set is the horizontal column hops.

Converting to Pauli terms

The first step needed to simulate H is to map the fermionic terms into *local* Pauli operations involving only a few qubits. When the Hamiltoinan acts on a 1D lattice, using the Jordan-Wigner would suffice. However, for the 2D lattice, either row or column hops will end up being non-local terms involving $O(N)$ qubits. To overcome this, we use the Verstraete-Cirac encoding to convert the repulsion and hopping terms into local Pauli operators involving at most 4 qubits at a time. This is done by adding

SAMPLE 4/4

Quantum Chemistry

- https://aka.ms/AQRE/DoubleFactorizedChemistry

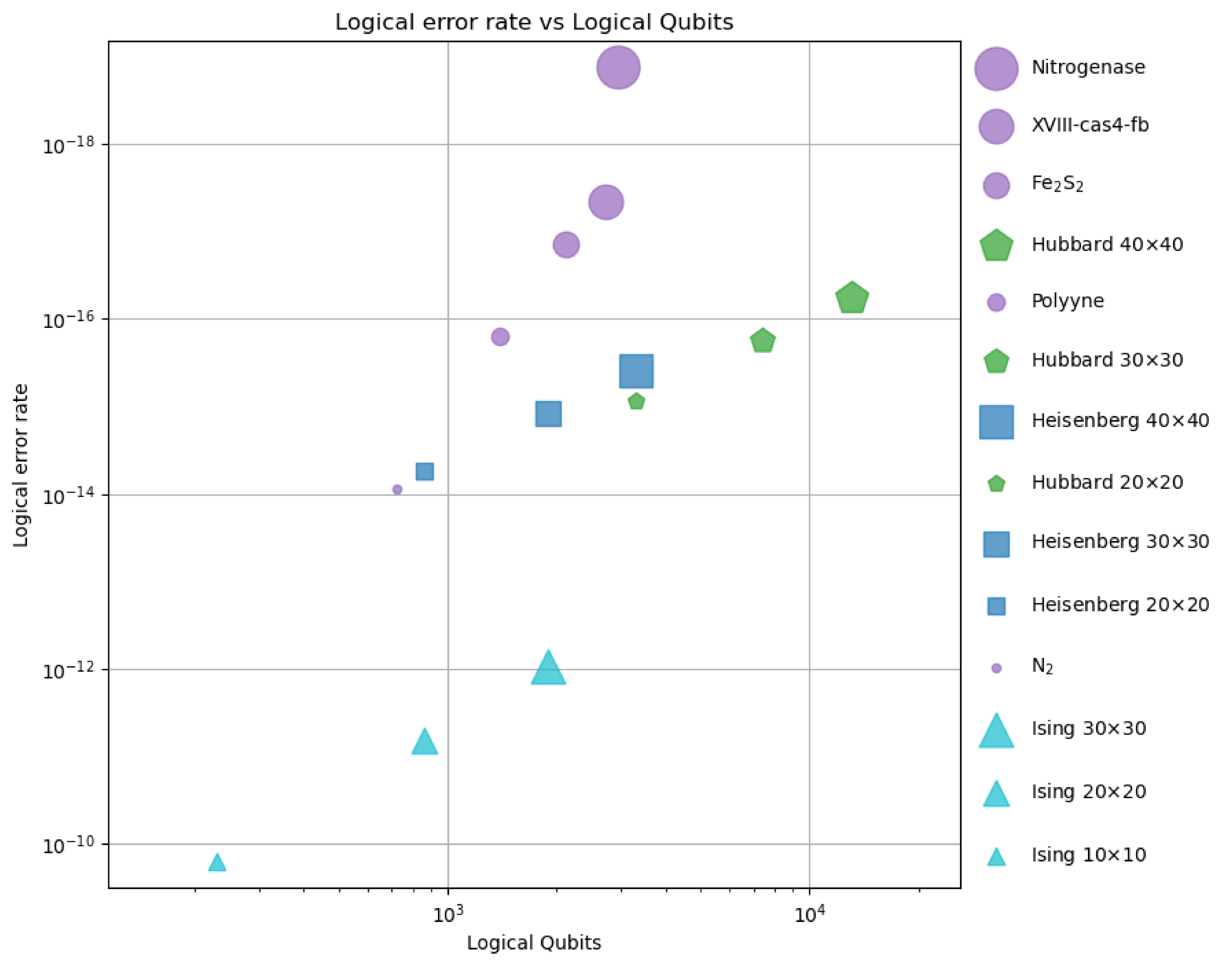
- Logical resource estimates for various molecules:

	N_2	Polyyne	Fe_2S_2	XVIII-cas4-fb	Nitrogenase
qubits	721	1394	2109	2740	2956
depth	5.7×10^8	1.6×10^{10}	1.2×10^{11}	2.7×10^{11}	8.6×10^{12}
error	8.6×10^{-15}	1.6×10^{-16}	1.4×10^{-17}	4.6×10^{-18}	1.3×10^{-19}
target rQOPS	3.2×10^5	1.7×10^7	2.0×10^8	6.0×10^8	2.1×10^{10}

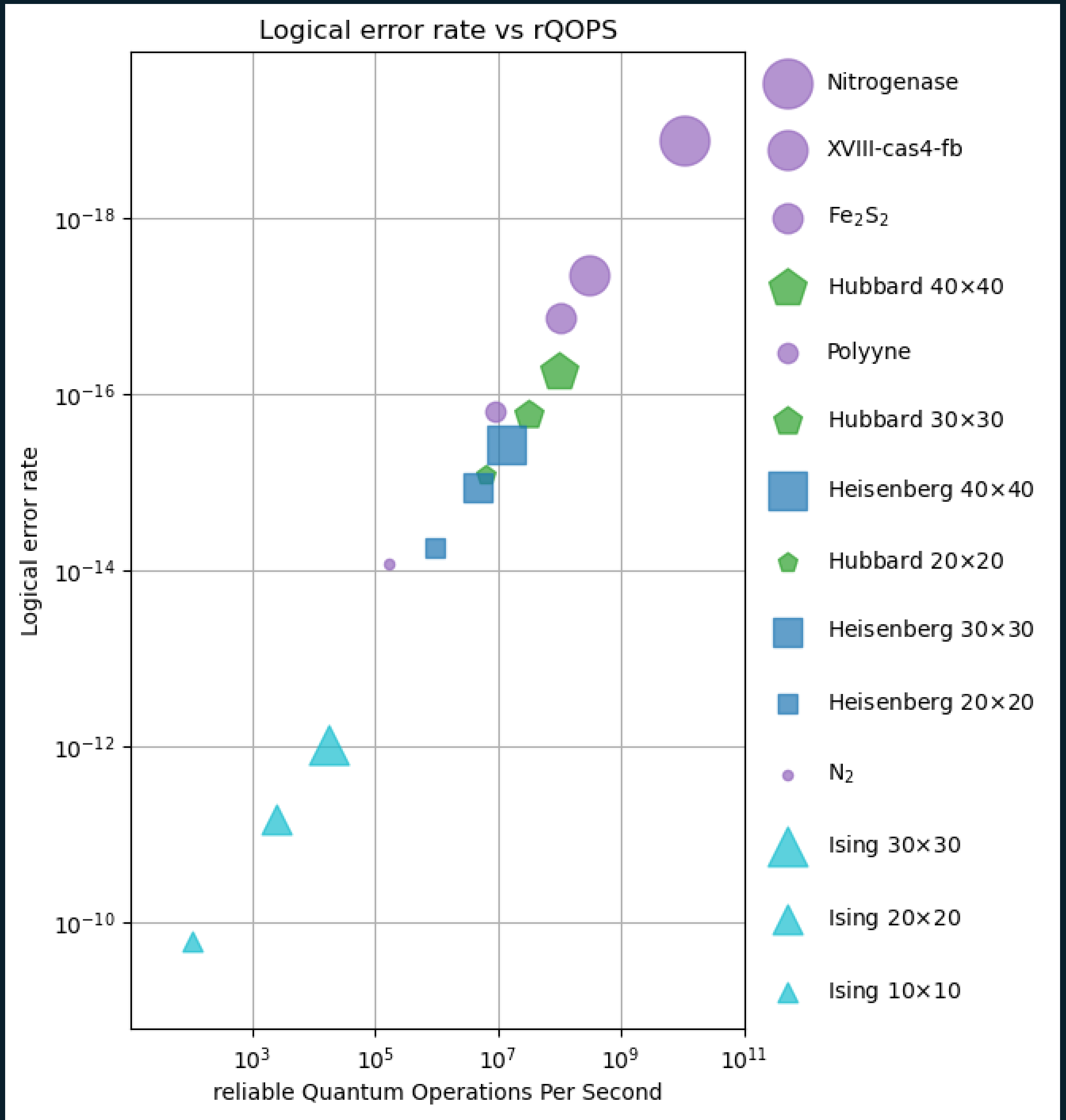
- Target run-time: 28 days

Logical Qubits vs Logical Errors

Ratio depends on algorithm

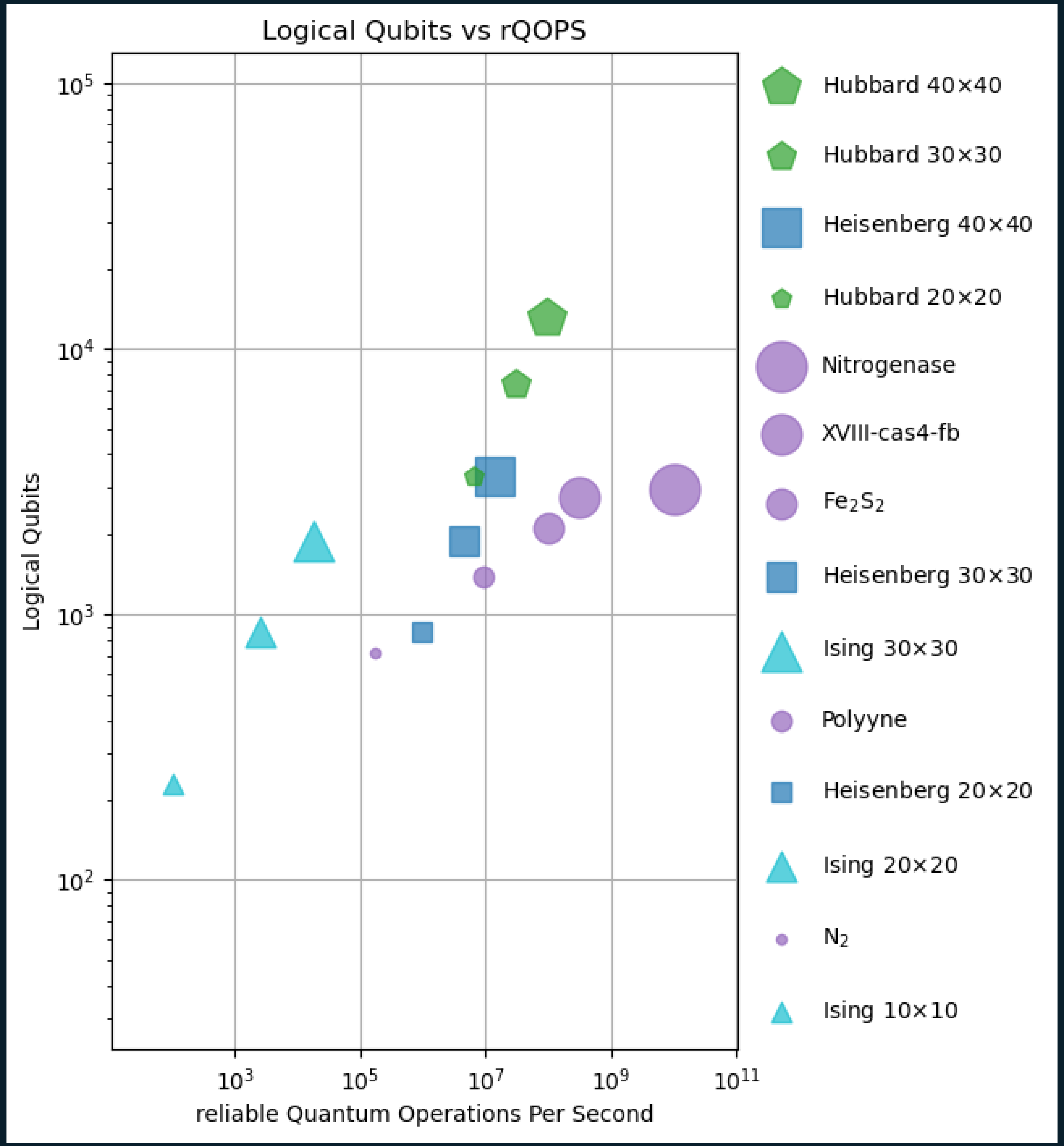


rQOPS vs Logical Qubits vs Logical Error Rates

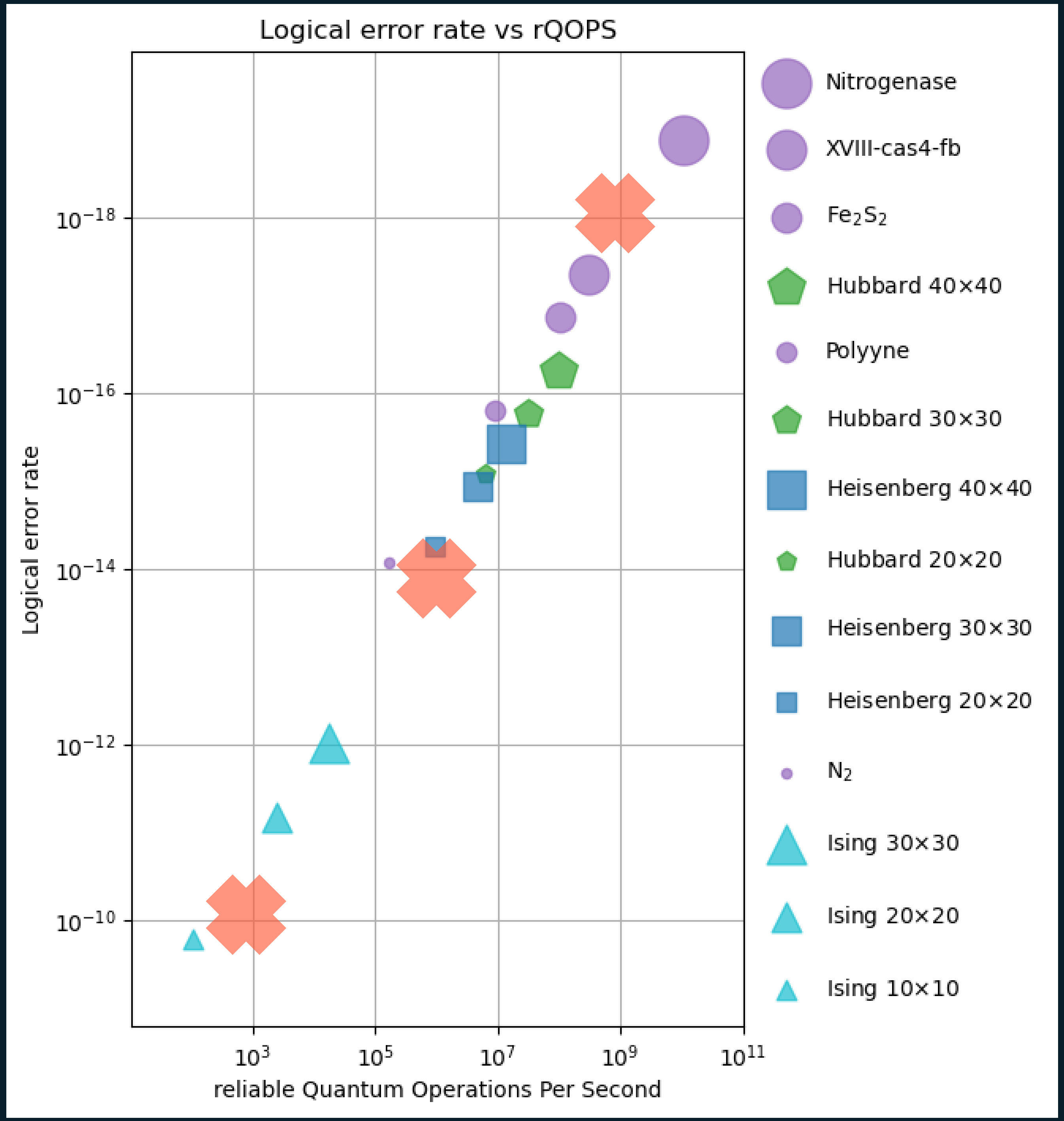


Larger systems:
Logical clock-speed matters most

Small systems:
Number of logical qubits matters most



Target Profiles for kilo, Mega, and Giga rQOPS



Giga rQOPS

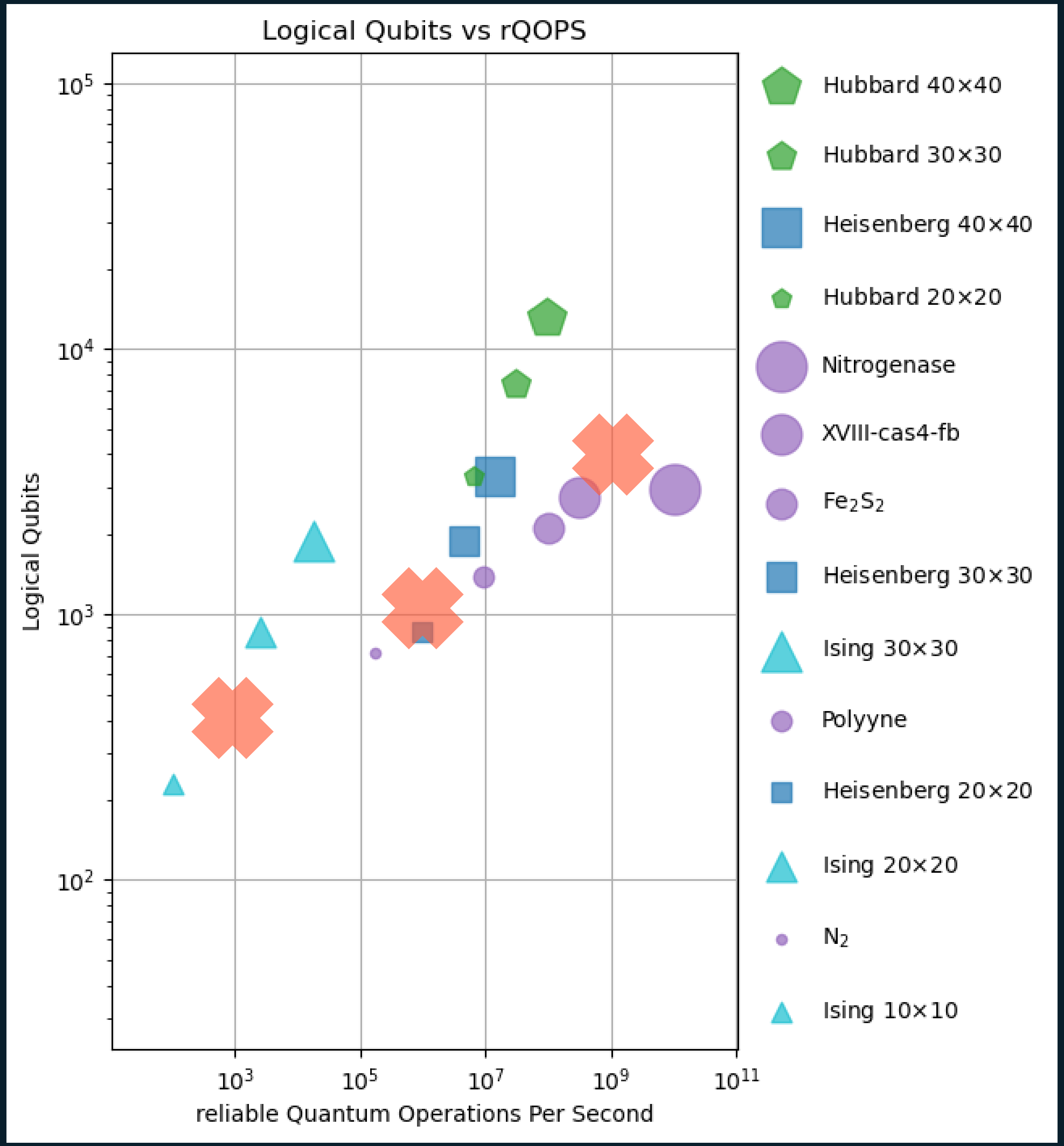
- 3000 logical qubits
- logical error ≤ 10^{-18}
- clock 3.3×10^{5} Hz

Mega rQOPS

- 1000 logical qubits
- logical error ≤ 10^{-14}
- clock-speed 10^{3} Hz

kilo rQOPS

- 400 logical qubits
- logical error ≤ 10^{-10}
- clock-speed 2.5 Hz



Estimating Logical Clock-Speed

$$f_{\mathrm{logical}} = \frac{f_{\mathrm{physical}}}{\mathrm{QEC-overhead}}$$

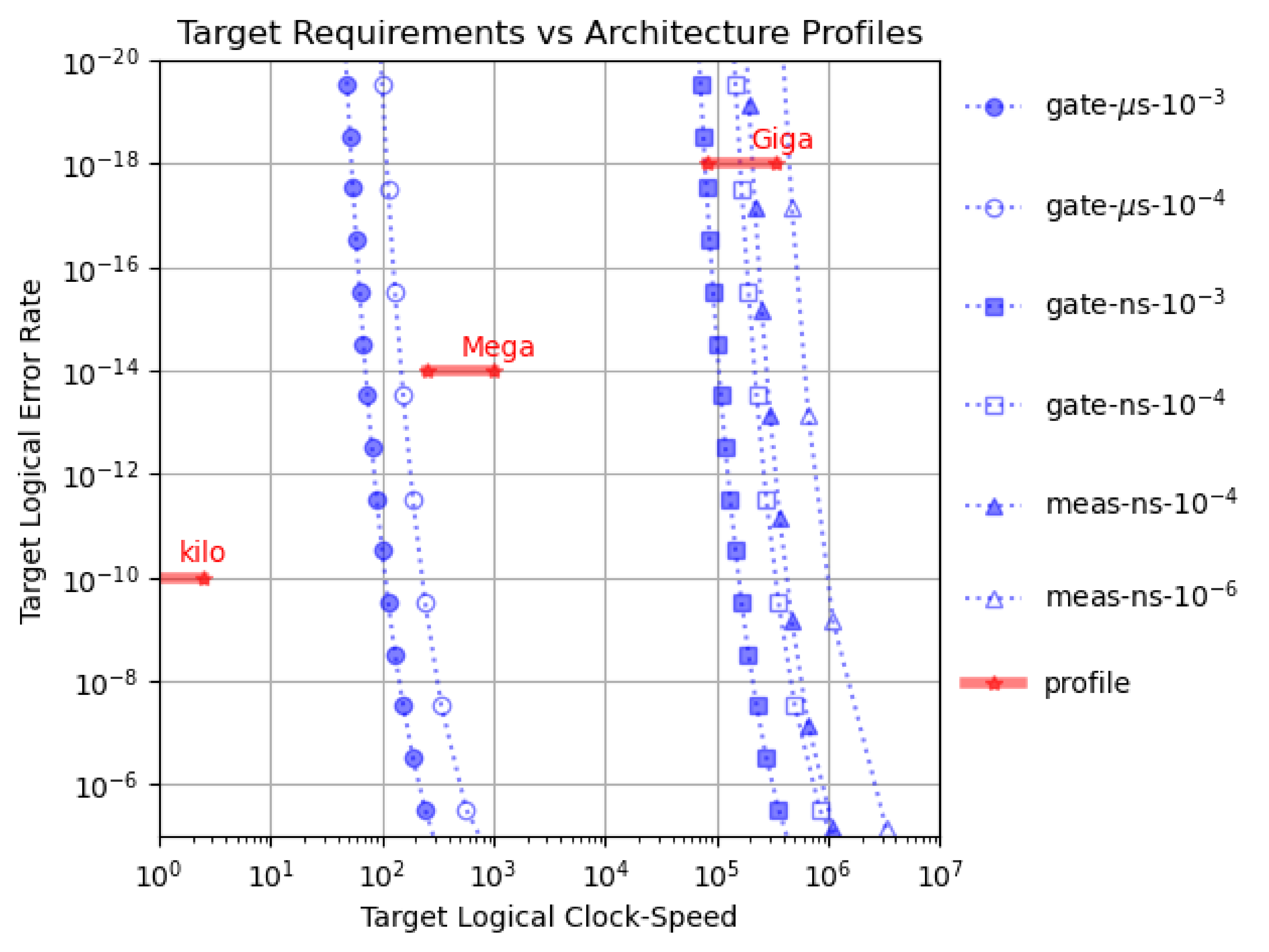
Logical clock-speed depends on
- physical clock-speed
- overhead induced by error correction
- and in turn target logical error rates

Architecture Profiles vs Target Profiles

kilo rQOPS:
Physical clock-speed not an issue

Mega rQOPS:
QEC slowdown starts to make a difference

Giga rQOPS:
Physical clock-speed + fast QEC crucial



Microsoft

Thank you

*Try it out: **aka.ms/AQ/RE***

*Read about AQRE: **arXiv:2311.05801***
*Learn background: **arXiv:2211.07629***