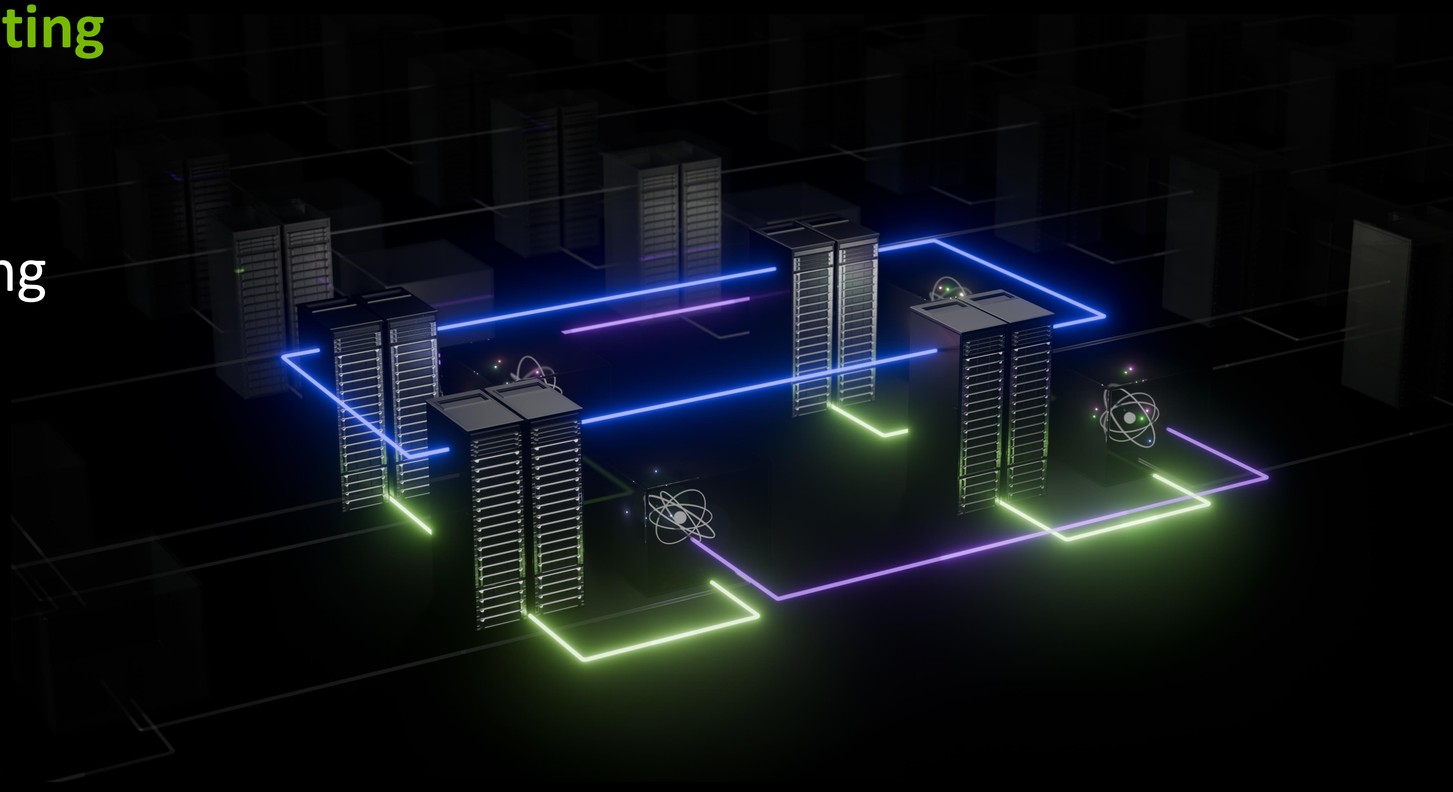
AIが生成する量子プログラムとその実行

Ikko Hamamura / NVIDIA

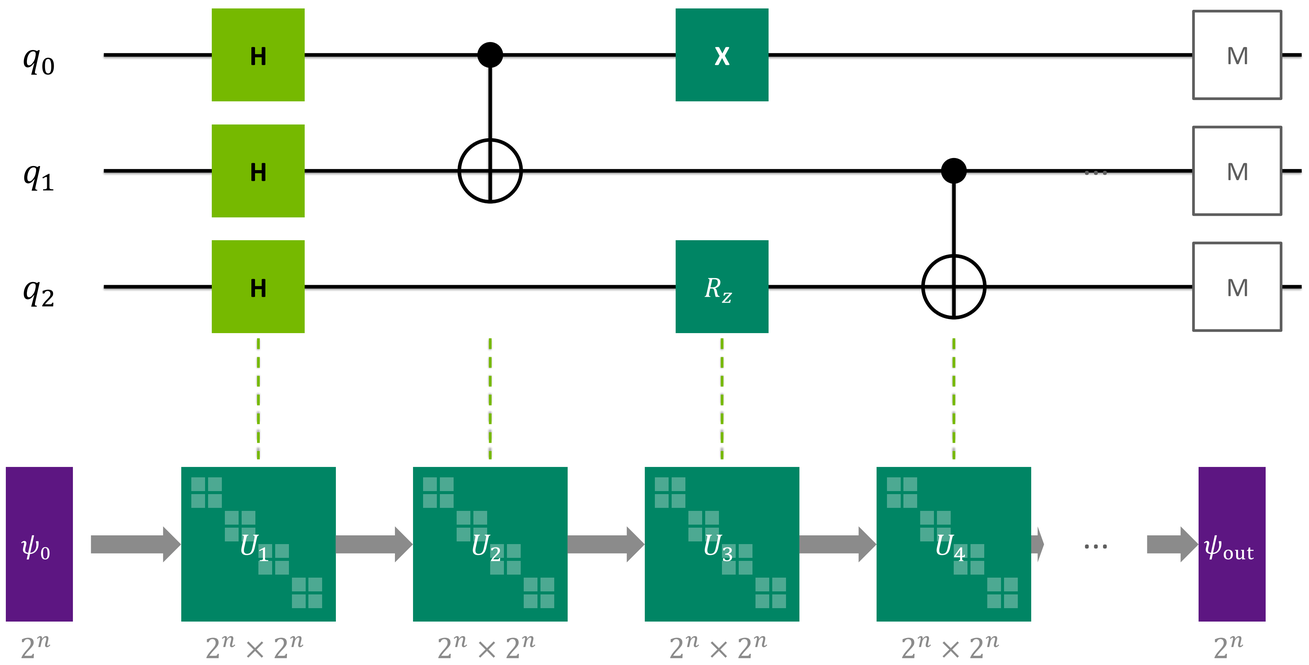
Quantum-GPU Supercomputing

- Supercomputing architecture **connecting quantum hardware**
- Ability to run **hybrid algorithms** - using GPUs and QPUs
- A **software platform** that seamlessly connects hybrid applications
- The ability to perform **qubit-agnostic** development of control and error correction



Quantum Circuits: Sparse Matrix-Vector Products

Quantum circuit (the program you write)



Linear algebra (what the machine does)

$$\psi_{\text{out}} = U_k \cdots U_2 U_1 \psi_0$$

Each U_i is sparse with 2^m nonzeros per row

k up to $\text{poly}(n)$ gates $\Rightarrow$ $O(k \cdot 2^n)$ flops in total

In Computational Terms

State — State vector $\psi \in \mathbb{C}^{2^n}$. At $n = 30$ that is $2^{30} \times 16\ \text{B} \approx 17\ \text{GB}$, and each extra qubit doubles it.

Gate — Each gate is a sparse unitary $U \in \mathbb{C}^{2^n \times 2^n}$ with only 2^m nonzeros per row for an m-qubit gate.

Circuit — A circuit of depth k is k sparse mat-vec products in sequence, with k up to $\text{poly}(n)$.

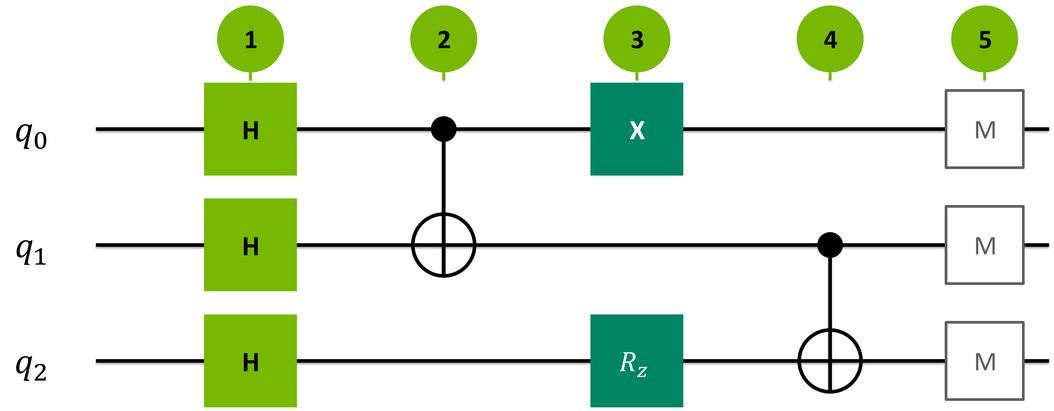
Output — Measurement returns a single index i with probability $\Pr(i) = |\psi_i|^2$.

Quantum Kernels: The Circuit as a Program

Quantum kernel written in CUDA-Q (Python)

```python
@cudaq.kernel
def circuit(theta: float):
    q = cudaq.qvector(3)        # state |000>
    h(q)                        # H on every qubit          1
    x.ctrl(q[0], q[1])          # CNOT q0 -> q1             2
    x(q[0]); rz(theta, q[2])    # X and Rz                  3
    x.ctrl(q[1], q[2])          # CNOT q1 -> q2             4
    mz(q)                       # measure all               5
counts = cudaq.sample(circuit, 0.3, shots_count=1000)
```

Quantum circuit drawn from the same program



Kernel line order = circuit column order = matrix product order

Device code

Written on the host and launched on a target, like a CUDA kernel. The body is a real program with arguments, function calls, loops and branches.

Qubit-agnostic

One source is compiled for the chosen hardware, whether a QPU of any qubit modality or a simulator such as state vector or tensor network.

Line = matrix

Each gate call applies one sparse $2^n \times 2^n$ unitary to the state, so the listing is $U_k \cdots U_1$ written out one factor per line.

sample = histogram

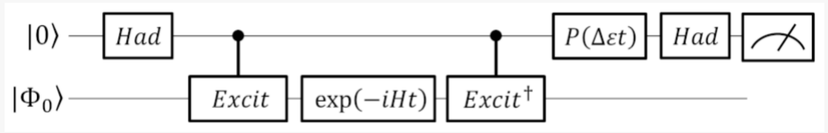
The kernel is launched 1000 times and each run returns one bitstring. The result is a histogram of outcomes. The 2^n vector entries never leave the device.

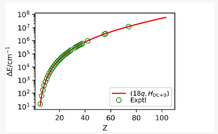
Quantum Algorithms: Time Evolution and Eigenvalue Problems

Time evolution

$$\psi(t) = e^{-iHt}\psi(0)$$

Quantum simulation

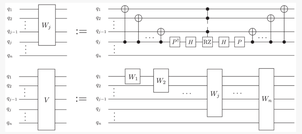


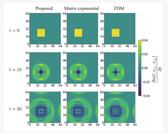


Energy gaps of boron-like ions, $Z = 5$ to 103, from Bayesian phase difference estimation on 18 qubits, simulated with cuQuantum.

Sugisaki et al., Electron. Struct. 5, 035006 (2023)

Quantum CAE





For the wave equation, finite-difference operators become explicit gates, giving a circuit for e^{-iHt} with depth $\mathrm{O}(\log_2 N)$ for N cells.

Sato et al., Phys. Rev. Research 6, 033246 (2024)

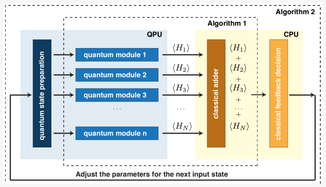
Eigenvalue problems

$$H\psi = E\psi$$

VQE

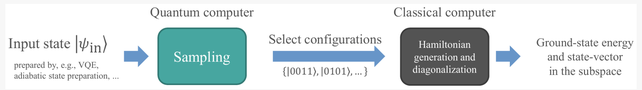
$$E_0 \approx \min_\theta \langle\psi(\theta)|H|\psi(\theta)\rangle$$

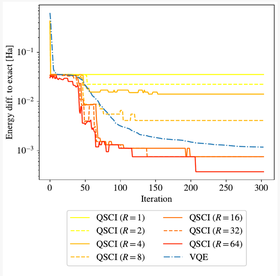
The QPU prepares $\psi(\theta)$ and measures the energy, a classical CPU/GPU optimizer updates θ.



Peruzzo et al., Nat. Commun. 5, 4213 (2014)

QSCI / SQD





Sample ψ on the QPU, keep frequent configurations, diagonalize H in that subspace on CPU/GPU. Variational and noise-robust.

Kanno et al., Phys. Rev. Research 8, 023268 (2026)

The Generative Quantum Eigensolver

First demonstration of GPT-generated circuits in the literature

Find a circuit producing e.g. ground state

Optimization of circuits for desired output

GPT-QE specifically employs a GPT model

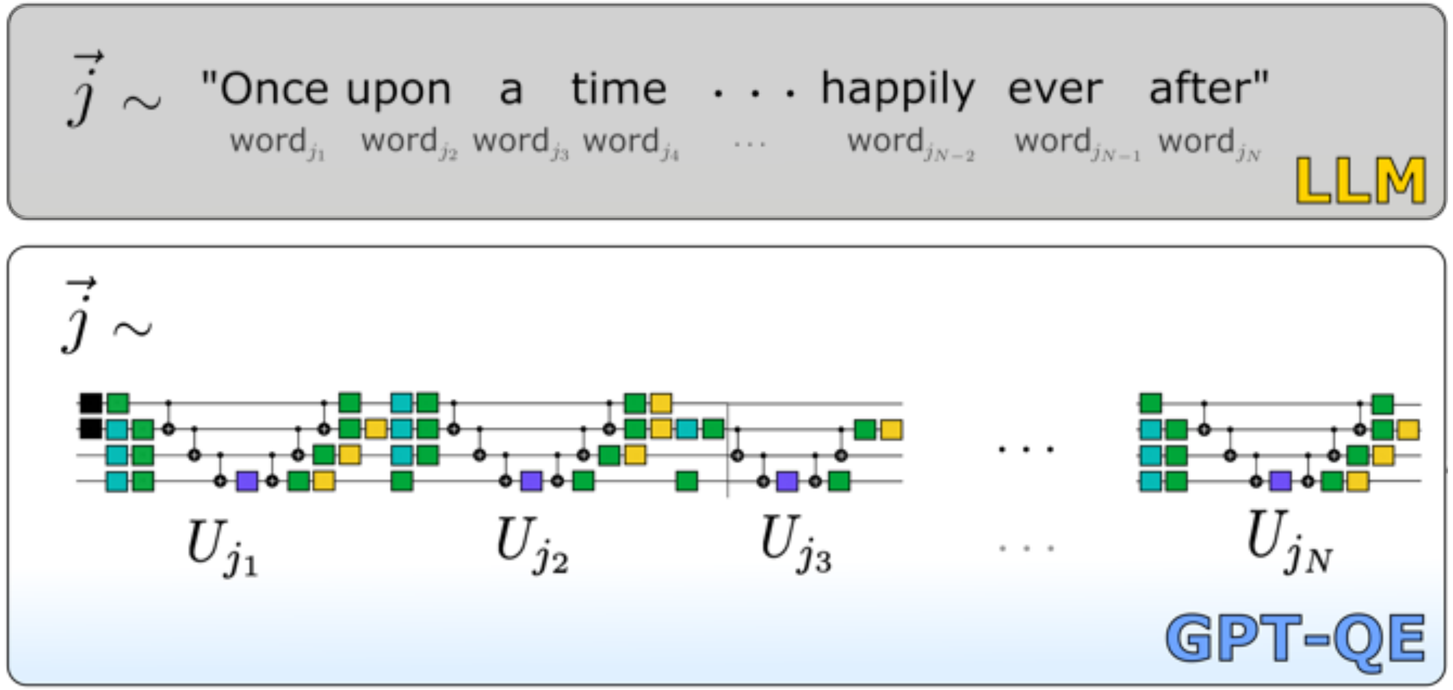
- Recent advances in attention-based transformer models can be leveraged

Analogy to Large Language Models (LLMs)

- Quantum operations are analogous to words
- Quantum circuits analogous to sentences
- When trained, GPT-QE generates a sequence of operations to form a circuit

Training learns weights in GPT model

- Cost function compares current prediction to measure of desired circuit output



https://arxiv.org/abs/2401.09253

Generative Circuit Design for QSCI

First GPT-designed circuits validated at 32-qubit scale

- QSCI needs **well-designed circuits** under hardware noise
- **GQE** — Transformer optimizes ansatz via QSCI energy
- **QSCI** — Samples determinants, diagonalizes in subspace

32 qubits

N_2 validated at 32-qubit scale

98% fewer

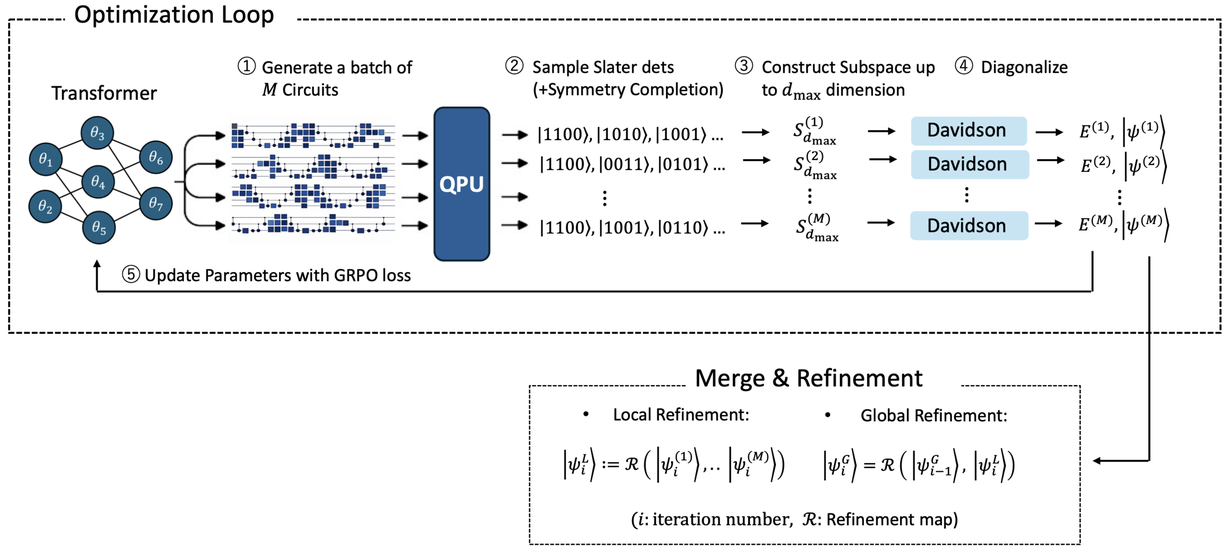
2Q gates vs. Trott er ansatz

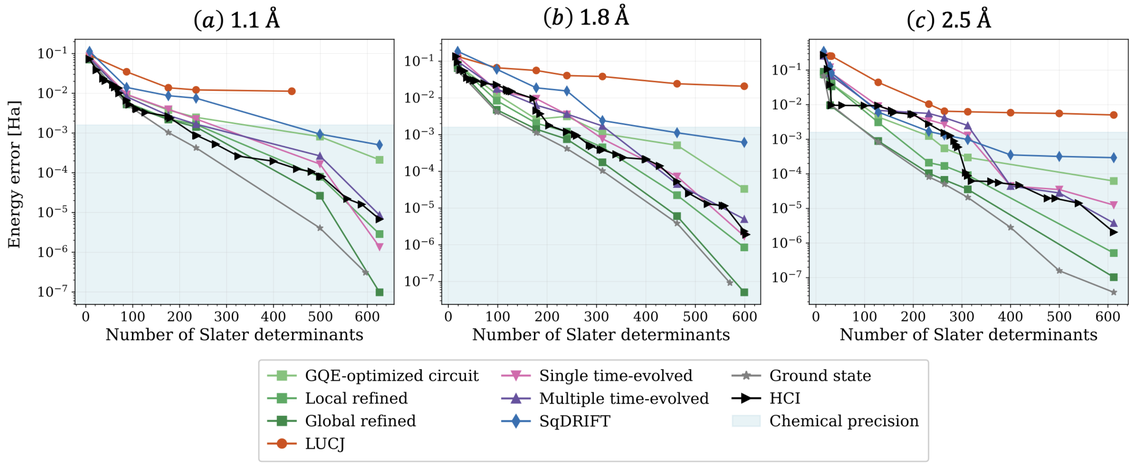
83% fewer

2Q gates vs. qDRIFT baseline

GQE + QSCI

Transformer optimizes ansatz energy





Keio University 1858 CALAMVS GLADIO FORTIOR

MITSUBISHI CHEMICAL GROUP
Mitsubishi Chemical

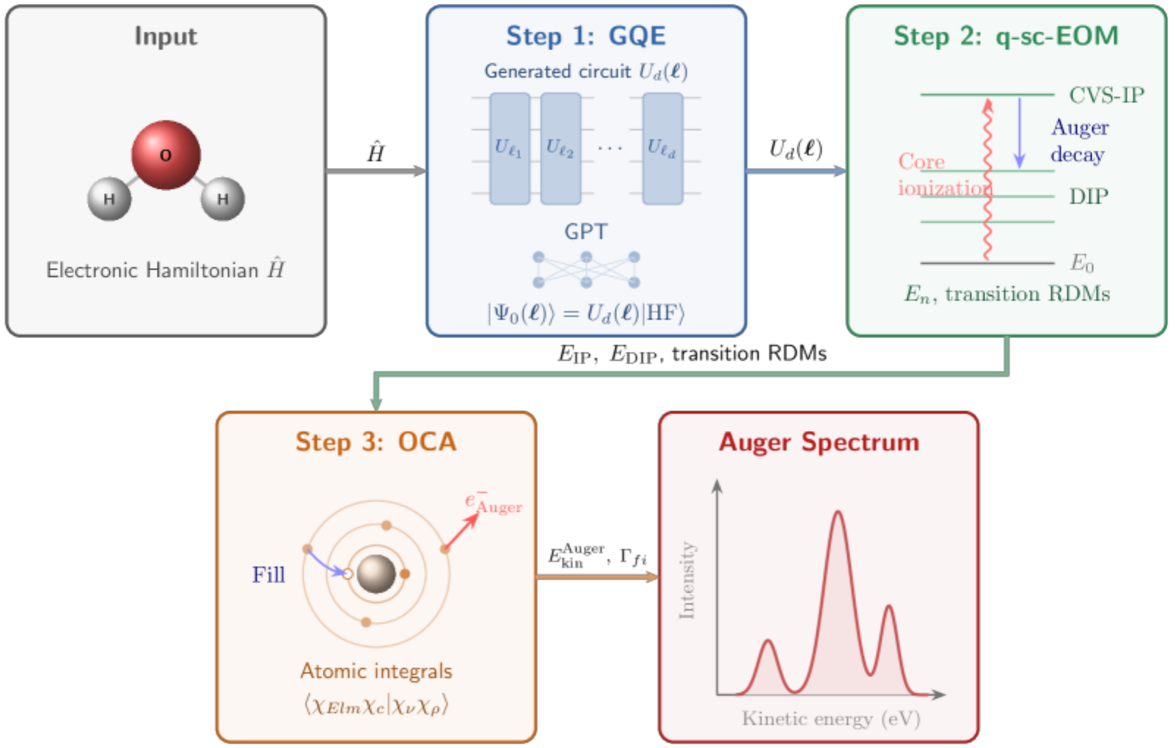
NVIDIA

Kemmoku et al., "Generative Circuit Design for Quantum-Selected Configuration Interaction," arXiv:2604.09756 (Apr 2026)

Auger Spectroscopy via GQE

Generated circuits match FCI and experiment for H_2O Auger spectra

- Auger spectroscopy needs **many excited states**
- **Hybrid quantum-classical**
  — GQE + q-sc-EOM + OCA
- GPU-accelerated via **CUDA-Q**
- Validated on **H_2O** — matches FCI & experiment
- **GQE halves gate count** vs. VQE



Mitsubishi Chemical

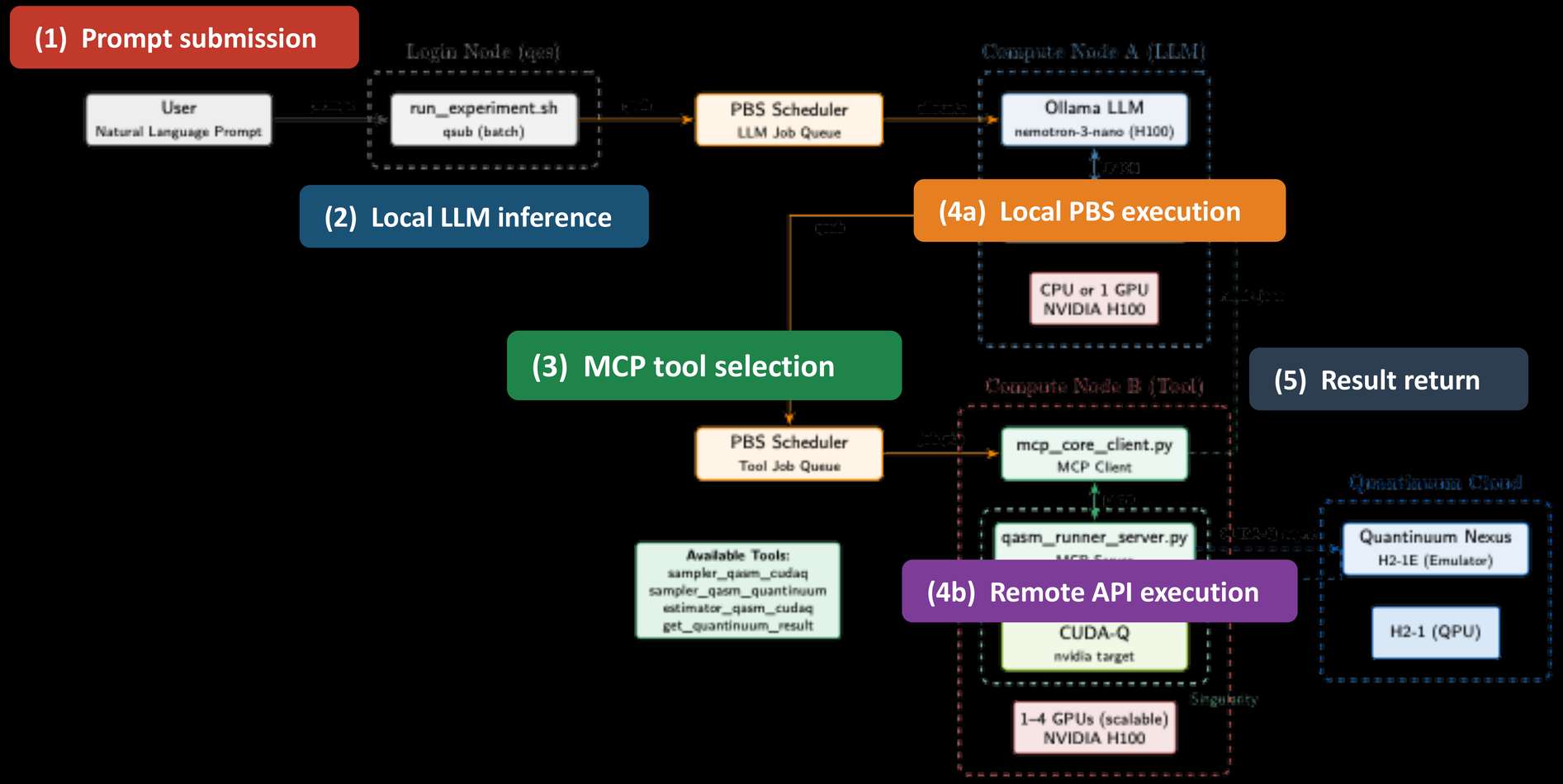
Keio University

NVIDIA

UNIVERSITY OF TORONTO

Keithley et al., “Auger Spectroscopy via Generative Quantum Eigensolver: A Quantum Approach to Molecular Excitations,” arXiv:2603.12859 (Mar 2026)

Our system: Local LLM + MCP + hybrid quantum backends



Local LLM inside ABCI-Q
MCP-based tool dispatch
Unified local / remote execution

Local deployment keeps execution inside ABCI-Q

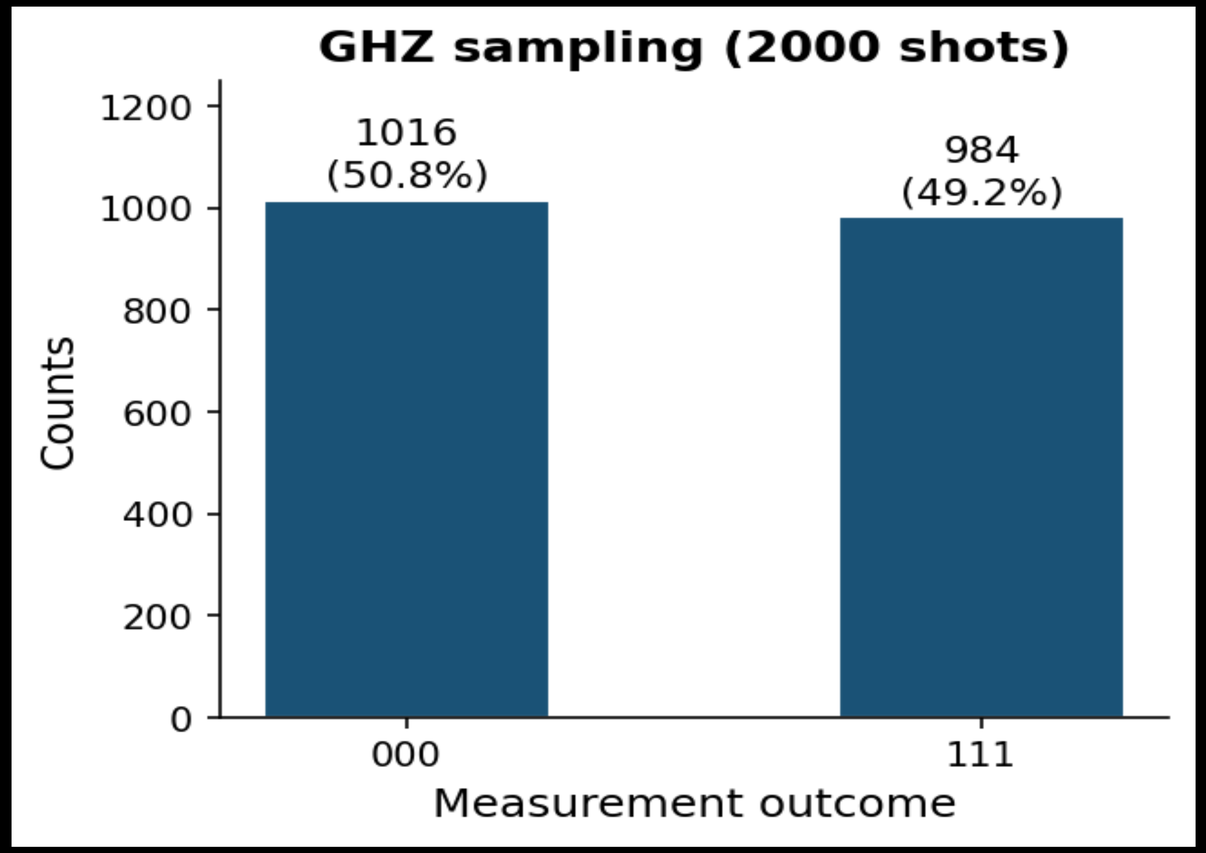
Shiraishi et al., “A Model Context Protocol Server for Quantum Execution in Hybrid Quantum-HPC Environments,” QCNC 2026, Kobe, Japan, pp. 1–6.

Result 1: Automated GHZ sampling on ABCI-Q Supercomputer (GPU Nodes*)

① Submit prompt

② LLM inference → MCP sampler tool

③ PBS job → CUDA-Q on NVIDIA H100



$|000\rangle \approx 0.50, \quad |111\rangle \approx 0.50$

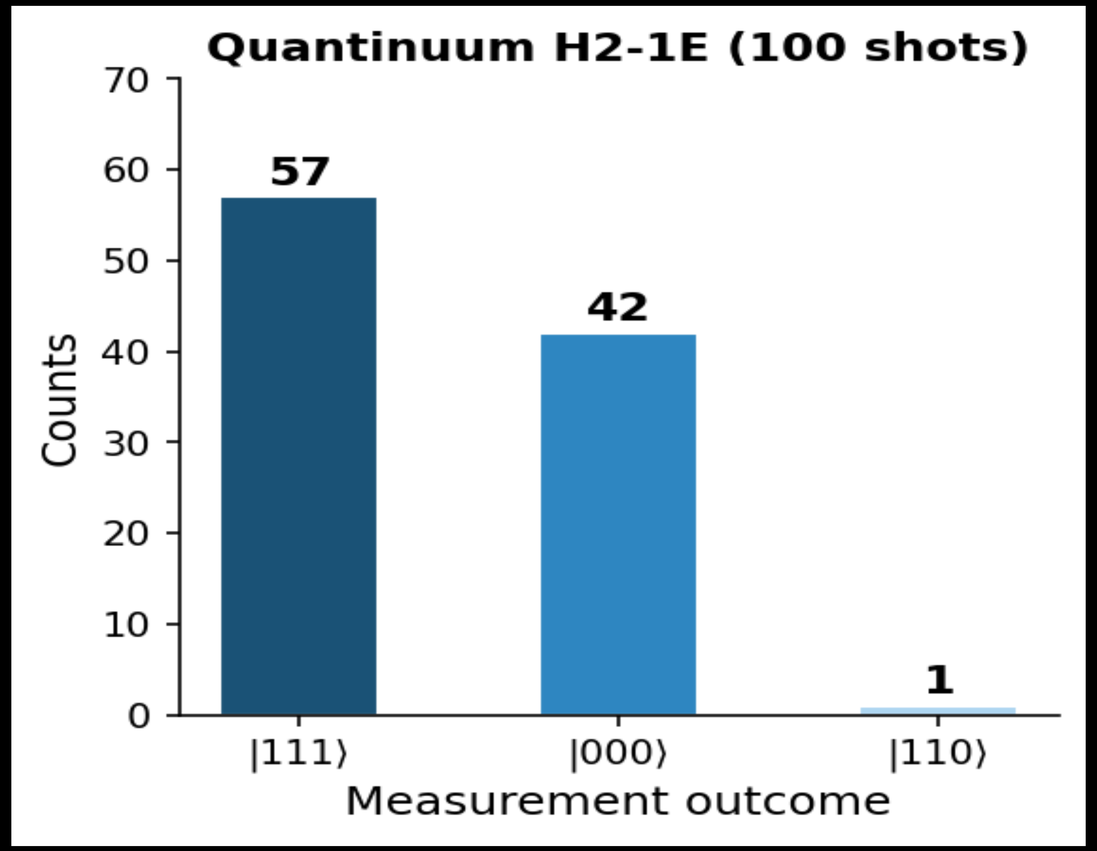
Shiraishi et al., “A Model Context Protocol Server for Quantum Execution in Hybrid Quantum-HPC Environments,” QCNC 2026, Kobe, Japan, pp. 1–6.

Result 2: Asynchronous execution on Quantinuum (Emulator)

① Submit job

② Receive job ID

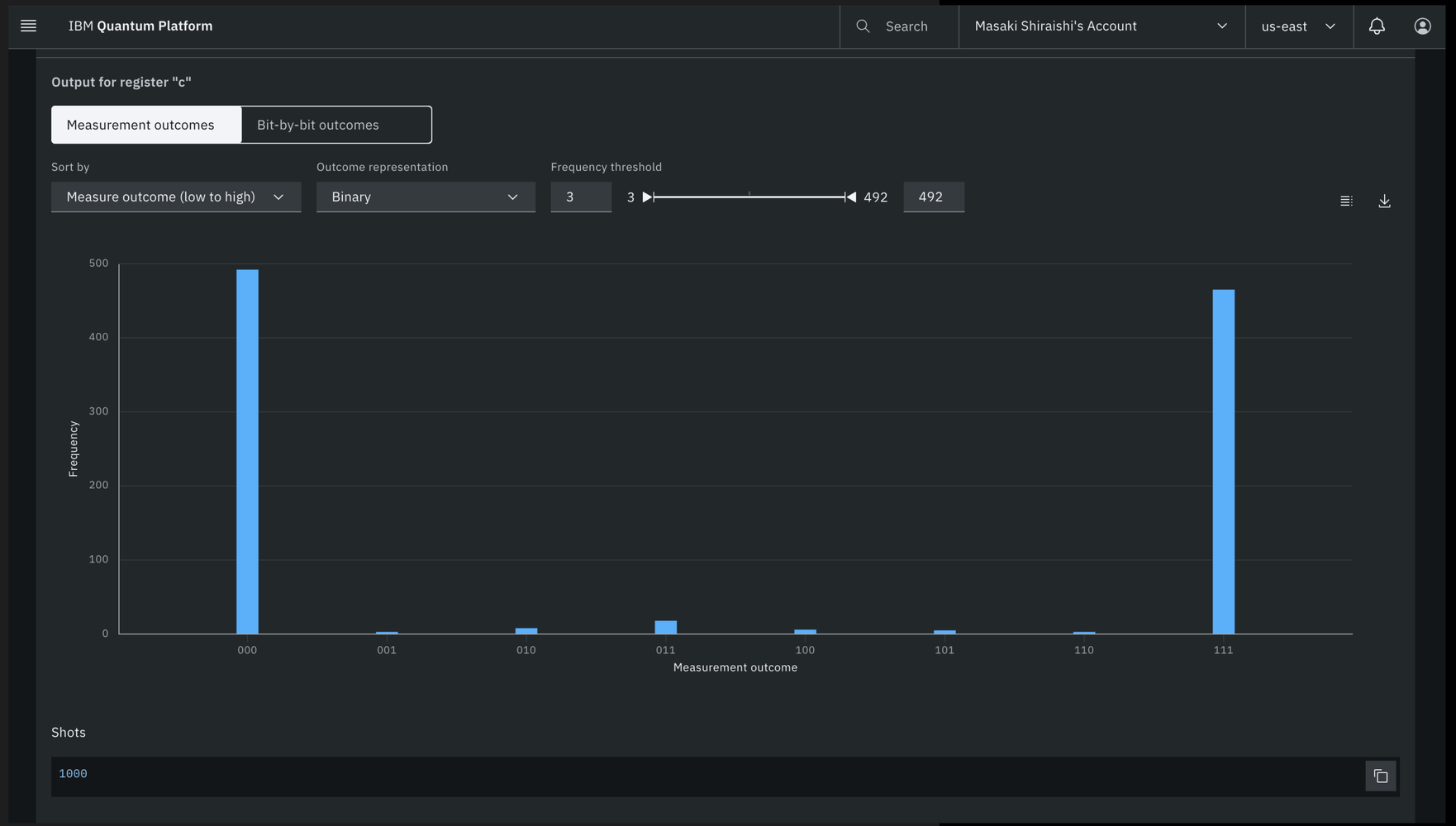
③ Retrieve result



Shiraishi et al., “A Model Context Protocol Server for Quantum Execution in Hybrid Quantum-HPC Environments,” QCNC 2026, Kobe, Japan, pp. 1–6.

Result 2’: IBM Quantum Platform from ABCI-Q

```
LLM resources: rt_QG=1, walltime=01:00:00
Tool resources: rt_QG=1, walltime=00:20:00

Submitted: 141325.qjcm

Waiting for job to complete...

=== LLM Job Started on qh238 ===
Job ID: 141325.qjcm
Date: Sat May 23 12:42:00 AM JST 2026

Starting Ollama server...
Ollama PID: 2991794
Ollama is ready (attempt 2)

=== Running MCP Host ===

=== Job Completed ===
Exit code: 0

Result:
{
  "job_id": "d887kk5g7okc73en70eg",
  "device": "ibm_kingston",
  "shots": 1000,
  "counts": {
    "000": 492,
    "111": 465,
    "010": 8,
    "011": 18,
    "110": 3,
    "100": 6,
    "101": 5,
    "001": 3
  },
  "probabilities": {
    "000": 0.492,
    "111": 0.465,
    "010": 0.008,
    "011": 0.018,
    "110": 0.003,
    "100": 0.006,
    "101": 0.005,
    "001": 0.003
  }
}
```



Shiraishi et al., “A Model Context Protocol Server for Quantum Execution in Hybrid Quantum-HPC Environments,” QCNC 2026, Kobe, Japan, pp. 1–6.

AI×量子