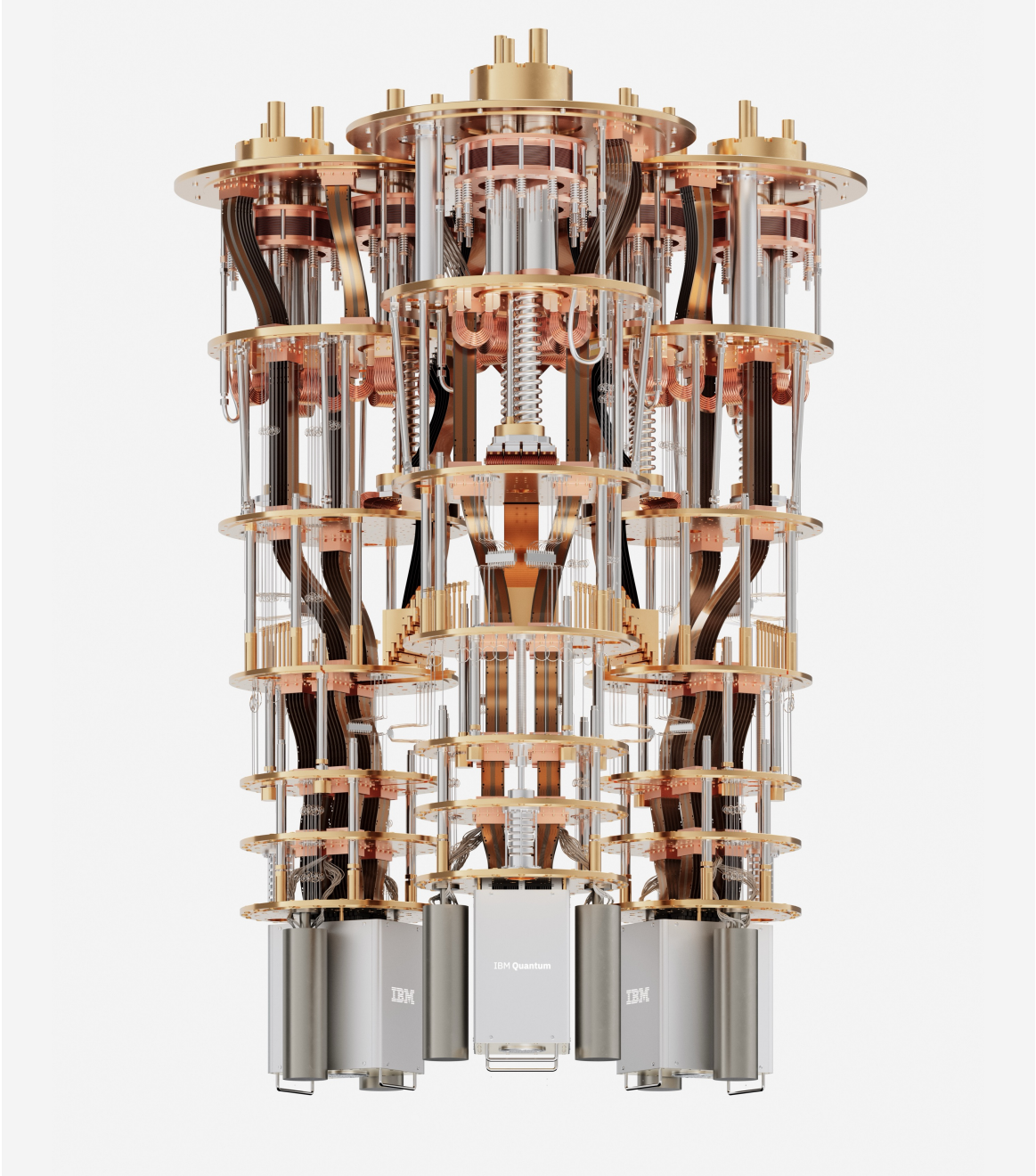
Workflows for Quantum-centric Supercomputing

Antonio Córcoles
Principal Research Scientist
Head of Quantum + HPC
IBM Quantum

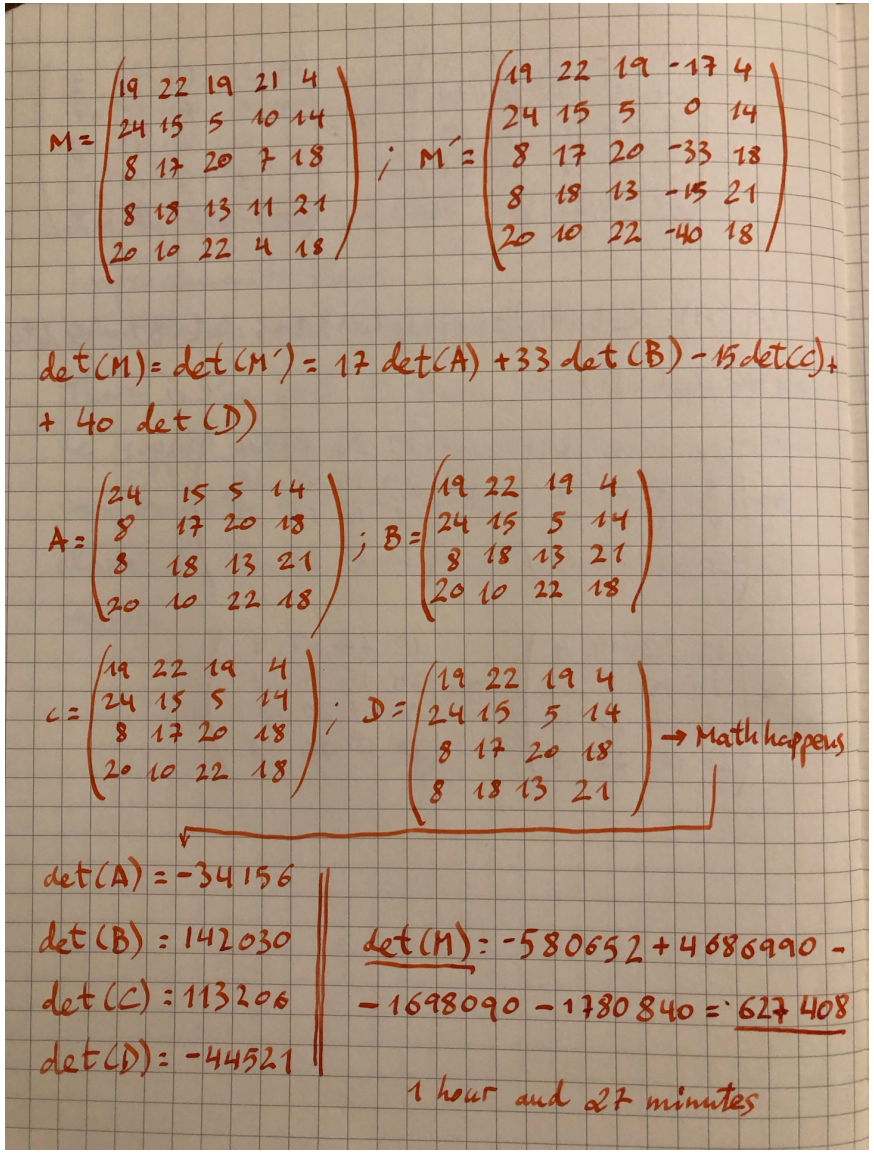


adcorcol@us.ibm.com
Qiskit Global Summer School 2024

Why do we use computers? – The determinant

A sub-routine for linear-systems solutions, and used extensively in vector calculus

Myself:



NumPy:

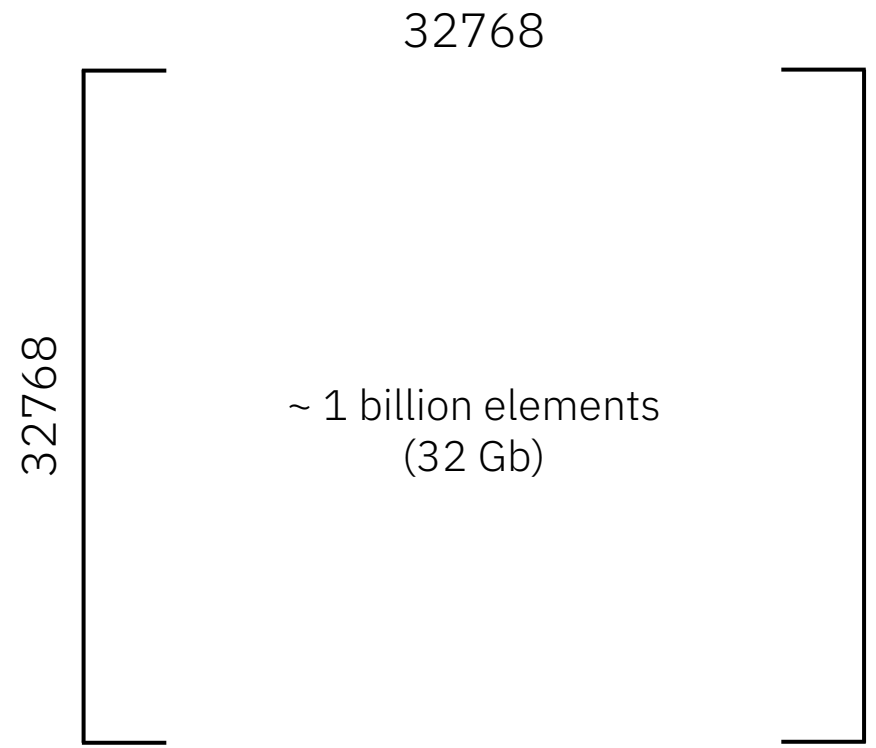
```
la.det(np.array([[19, 22, 19, 21, 4],
                 [24, 15, 5, 10, 14],
                 [8, 17, 20, 7, 18],
                 [8, 18, 13 11, 21],
                 [20, 10, 22, 4, 18]]))
```

Answer: 597888.0

Solution time ~ 200 us (20 million times faster)

Why do we use computers? – The determinant

```
[[19, 22, 19, 21, 4],
 [24, 15, 5, 10, 14],
 [8, 17, 20, 7, 18],
 [8, 18, 13 11, 21],
 [20, 10, 22, 4, 18]]))
```



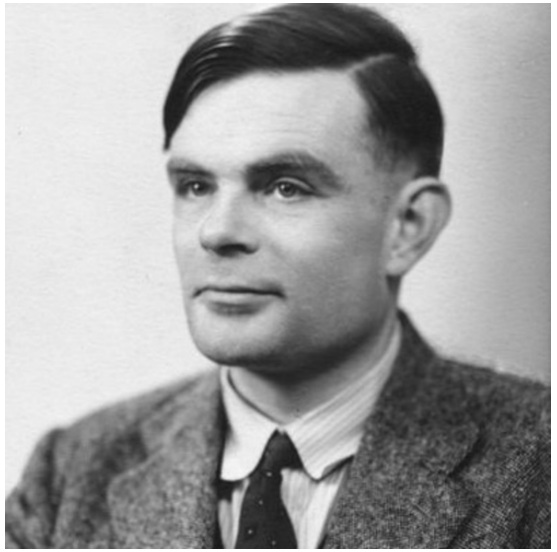
For common, everyday computing tasks, classical computations can scale without the loss of accuracy

What is computing?

We can think of a computer in the following way:

- The computer has an initial state.
- The state evolves following a finite sequence of operations.
- There is a mechanism to extract information on the state.

This is essentially a simplified model for a Turing machine.



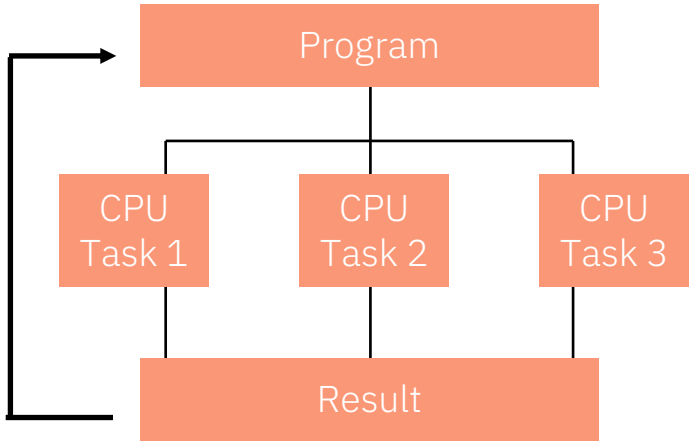
Alan Turing is considered the father of theoretical computer science

What is High-Performance Computing?

(aka HPC, Supercomputing, Extreme Scale Computing)

Serial Computing:

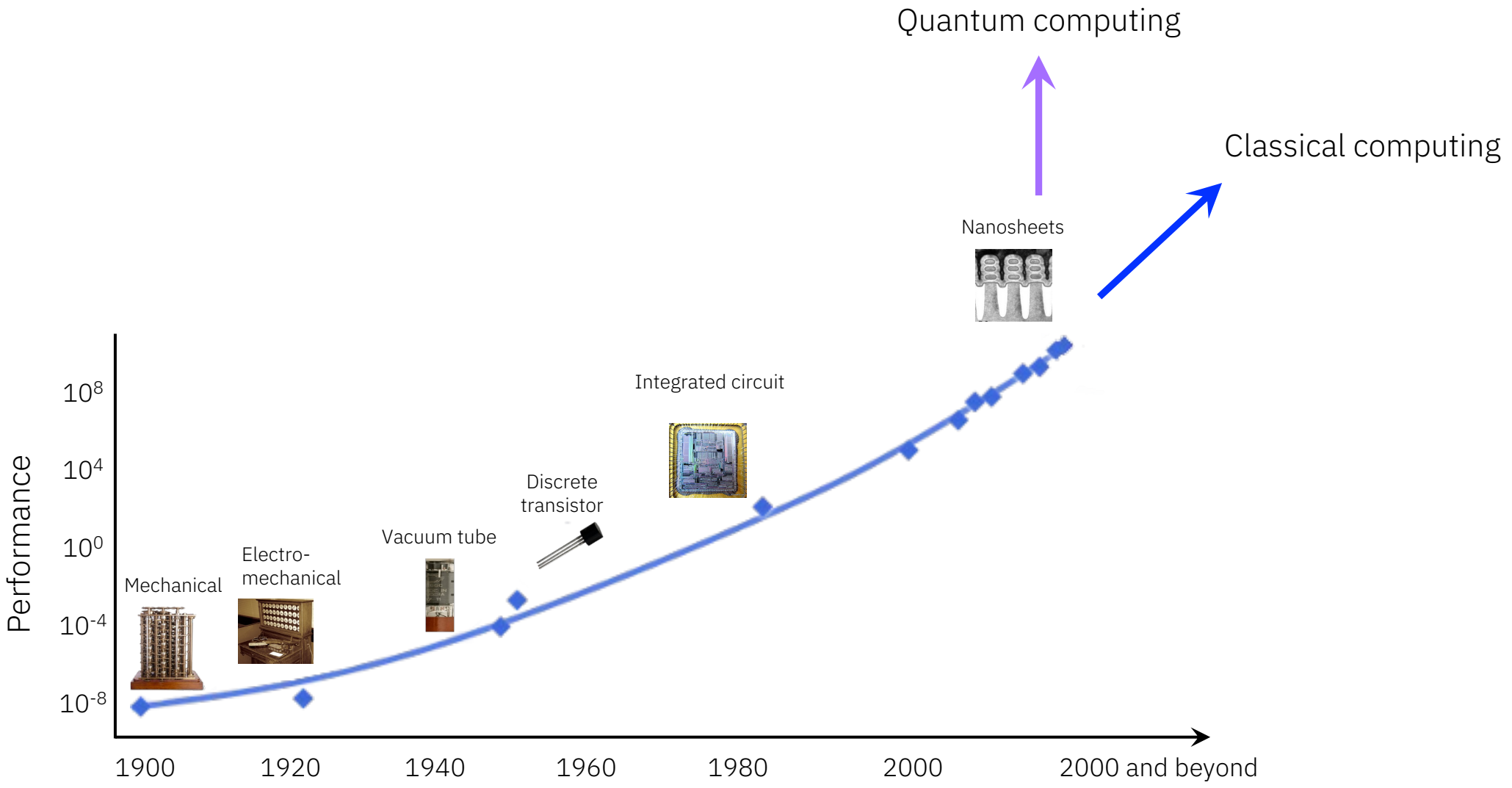
Parallel Computing:



HPC comprises:

1) Massively Parallel Computing
2) Computer Clusters
3) High-Performance Components

The paradigm of computing is branching



What is quantum computing?

We can use the same model for quantum computers as we used for the Turing machine , but the rules for these key points will be different:

1. a physical system in a perfectly definite state can still behave randomly

2. two systems that are too far apart to influence each other can nevertheless behave in ways that, though individually random, are somehow strongly correlated.

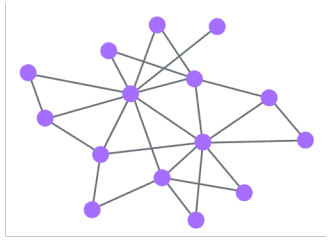
Richard Feynman proposed the idea of a quantum computer

What are Qiskit Patterns?

A guide for writing workflows to take advantage of quantum computation

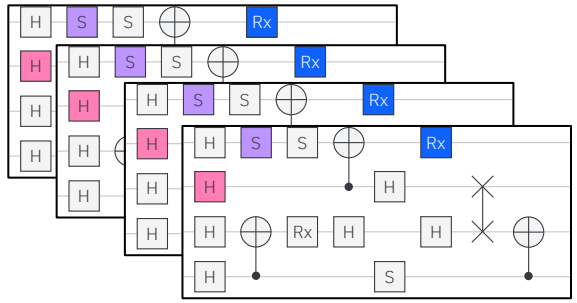
A quantum programming model

Map classical inputs to a quantum problem.



(abstract circuits, observables)

Optimize problem for quantum execution.



(ISA circuits, observables)

Execute using Qiskit Runtime Primitives.

circuit $(\vec{\theta})$

Sampler →

```
000101...,
110110...
```

bit-strings

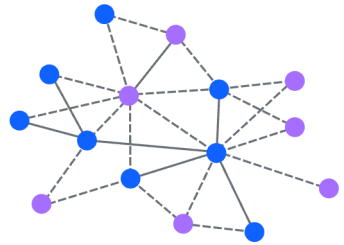
circuit $(\vec{\theta})$
+ observable $\hat{O}$

Estimator →

$\langle \hat{O} \rangle$

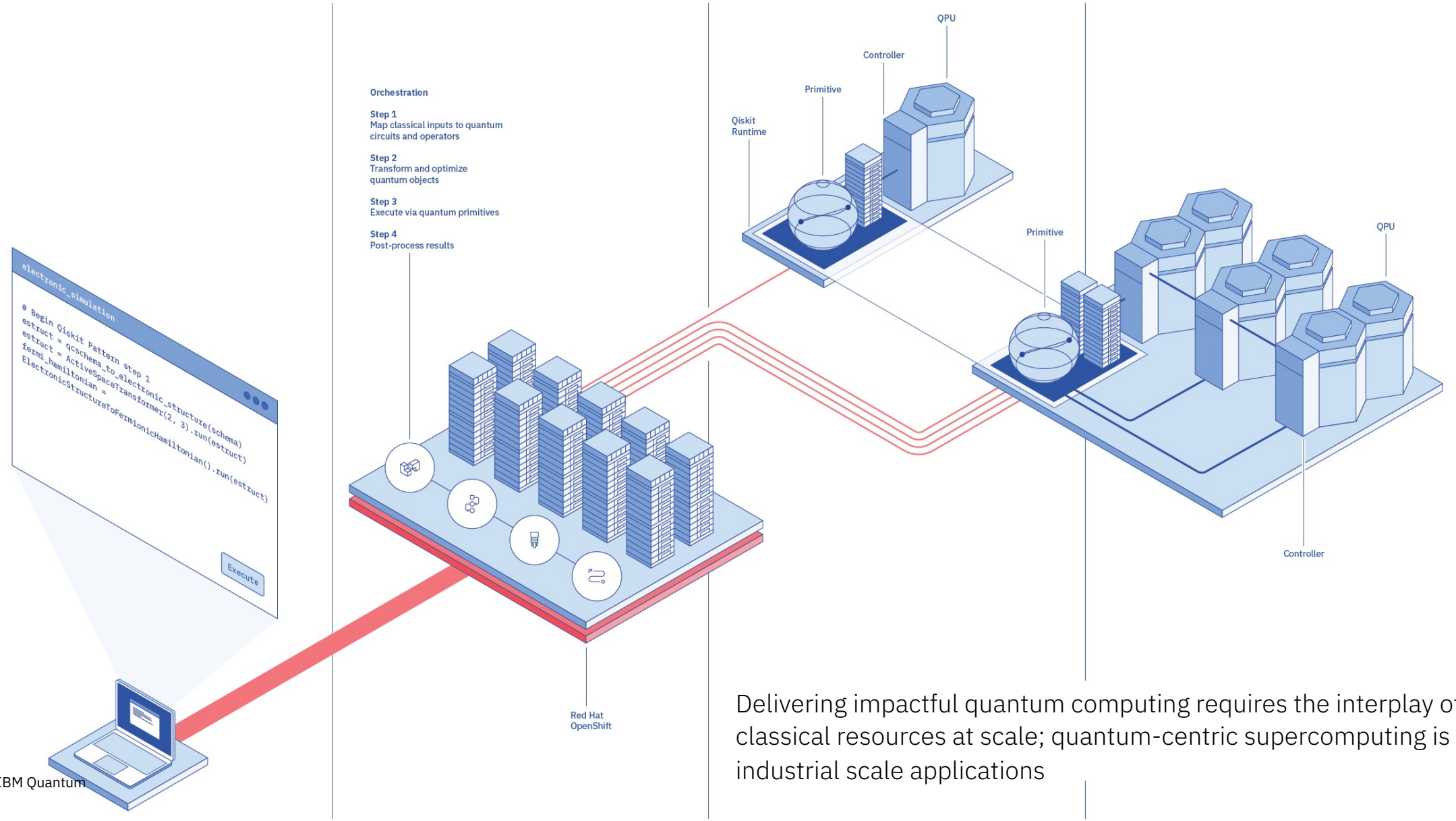
expectation value

Postprocess, return result in classical format.

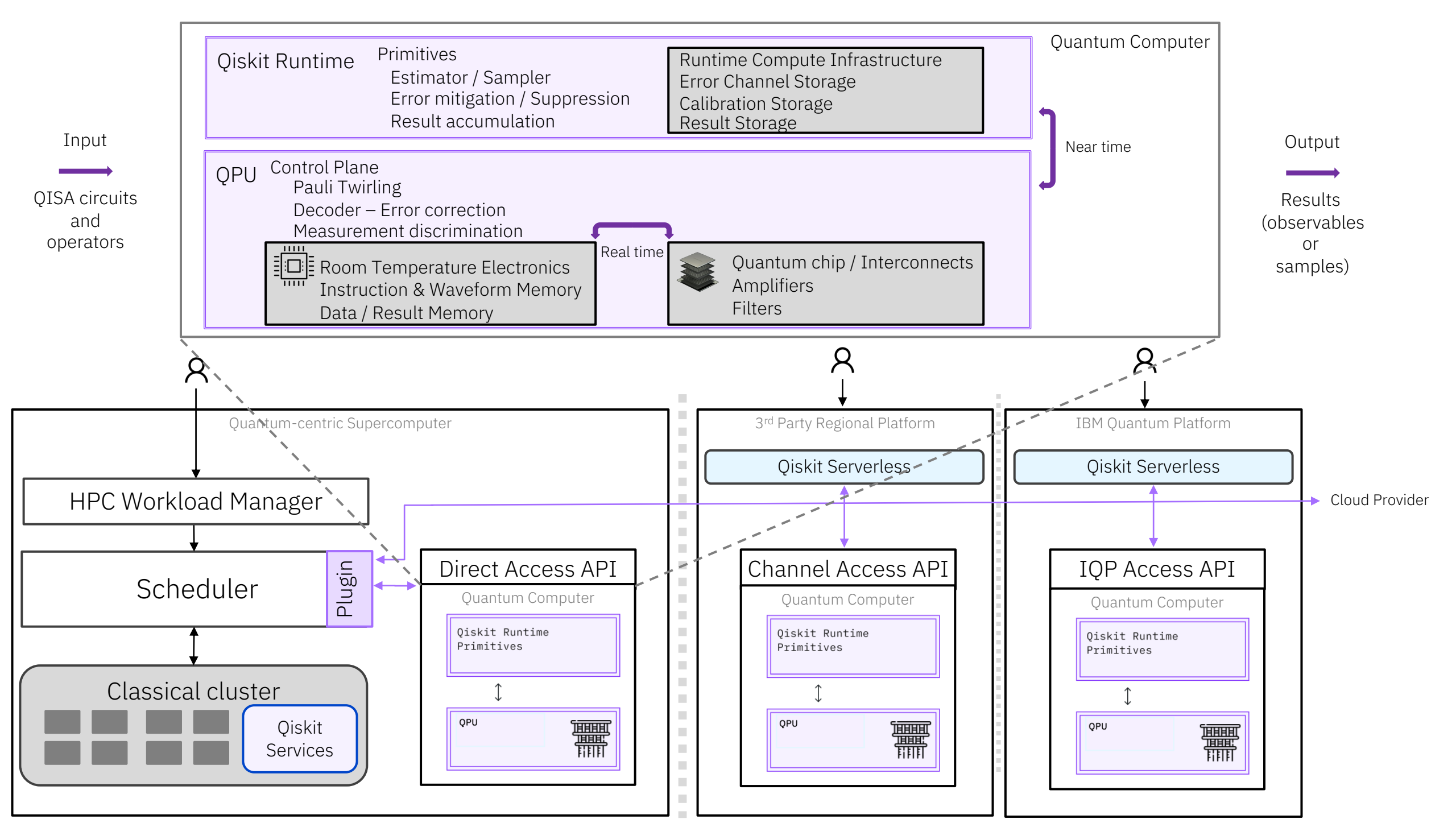


All steps are amenable to parallelization

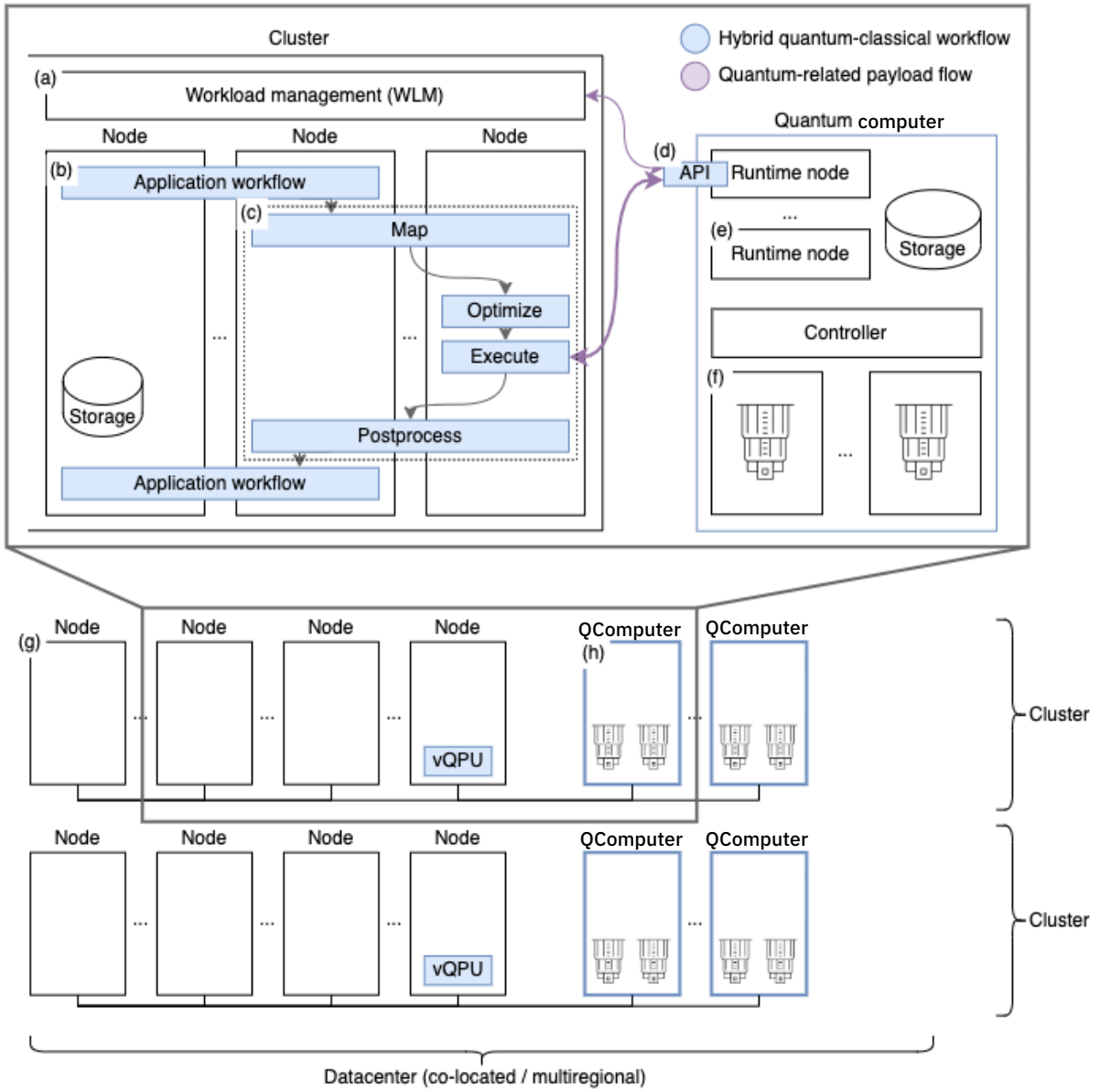
What is Quantum-centric supercomputing?



Delivering impactful quantum computing requires the interplay of quantum and classical resources at scale; quantum-centric supercomputing is the path toward industrial scale applications


Quantum Computer
Qiskit Runtime
Primitives
Estimator / Sampler
Error mitigation / Suppression
Result accumulation
Runtime Compute Infrastructure
Error Channel Storage
Calibration Storage
Result Storage
Near time
Input
QISA circuits and operators
QPU
Control Plane
Pauli Twirling
Decoder – Error correction
Measurement discrimination
Room Temperature Electronics
Instruction & Waveform Memory
Data / Result Memory
Real time
Quantum chip / Interconnects
Amplifiers
Filters
Output
Results (observables or samples)
Quantum-centric Supercomputer
HPC Workload Manager
Scheduler
Plugin
Classical cluster
Qiskit Services
3rd Party Regional Platform
Qiskit Serverless
IBM Quantum Platform
Qiskit Serverless
Cloud Provider
Direct Access API
Quantum Computer
Qiskit Runtime Primitives
QPU
Channel Access API
Quantum Computer
Qiskit Runtime Primitives
QPU
IQP Access API
Quantum Computer
Qiskit Runtime Primitives
QPU

The workflow



(a) - WLM
(b) - Application workflow
(c) - Qiskit Pattern
(d) - Direct API endpoint
(e) - Classical system node
(f) - QPU
(g) - Classical node
(h) - Quantum computer

Circuit Knitting

A protocol whereby a quantum computational problem (1 circuit & 1 Pauli group) is broken down into multiple quantum computational problems whose outputs are then post-processed (knit) asynchronously using classical computing

Examples

- Embedding
- Circuit cutting
- Entanglement forging
- Operator Back-propagation
- Regev's version of Shor's algorithm
- Multi-product Formulas

Circuit knitting original definition:

Techniques that break down larger circuits into smaller circuits to run on a quantum computer, and then knit the results back together using a classical computer

Circuit Knitting: Embedding

Background: Variational Quantum Eigensolver

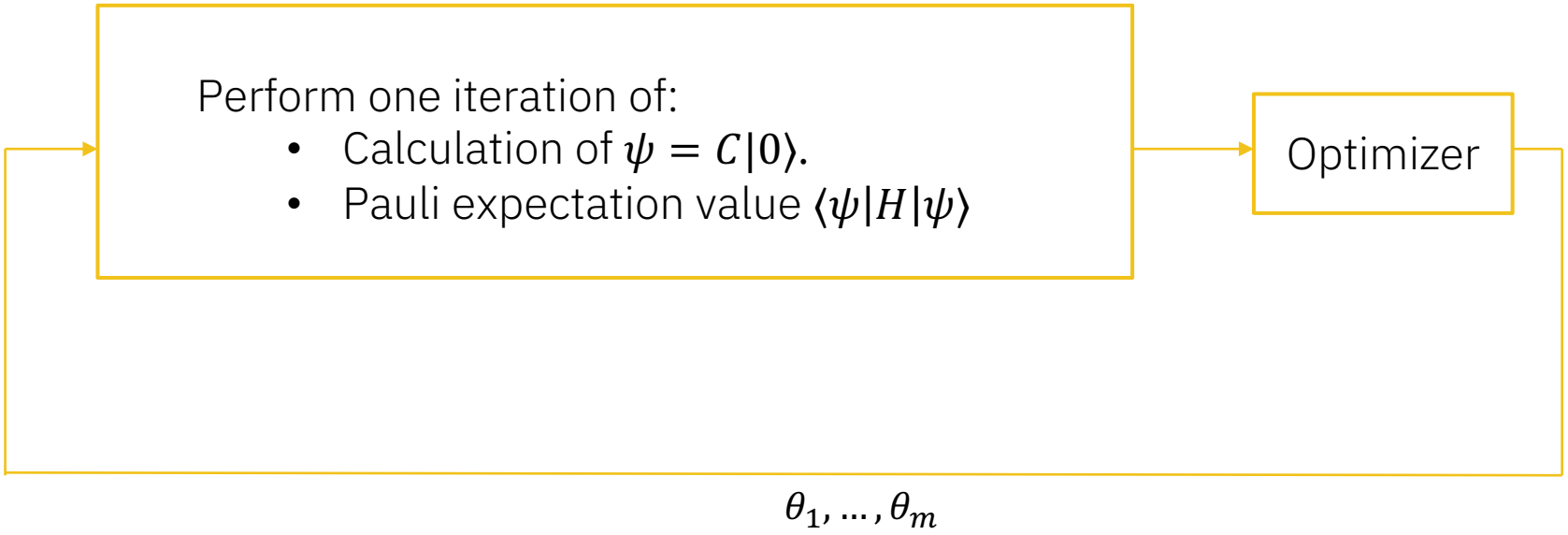
VQE Components

Hamiltonian H, represented as a weight of Pauli observables, eg.
$0.2 * IXYII - 1.6 * XZXII + 0.7 * YZIYX$

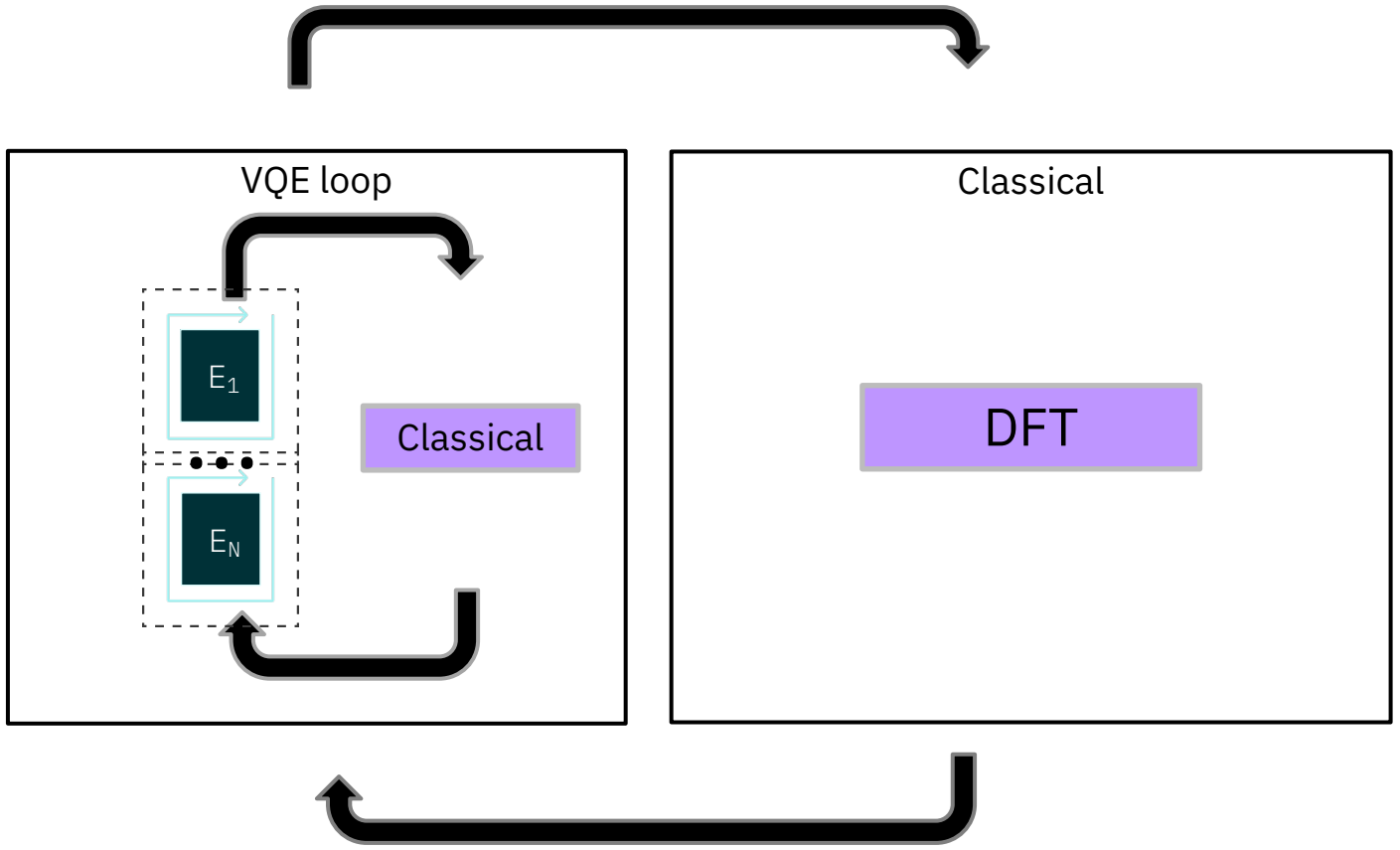
Goal: Find $|\psi\rangle$ that minimizes $\langle\psi|H|\psi\rangle$

Variational form circuit C, depends on parameters $\theta_1, \ldots, \theta_m$

Optimizer

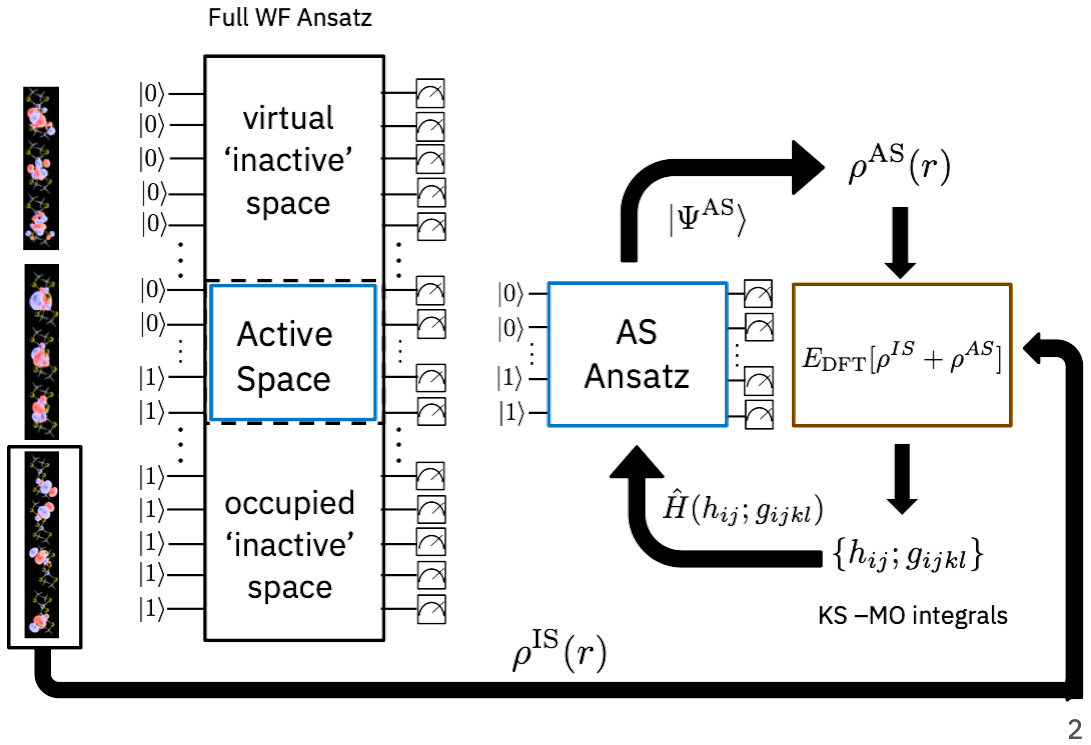


Circuit Knitting: Embedding



Embedding schemes compute only one part of the full system representation on the quantum computer

Circuit Knitting: Embedding



The quantum algorithm is restricted to a subset of active orbitals defined by an active space where the correlations are essential to the problem

Rossmannek *et al. J. Chem. Phys*. **154**, 114105 (2021)

Rossmannek *et al. J. Phys. Chem. Lett.* **14**, 3491 (2023)

Battaglia *et al.* arXiv:2404.18737 (2024)

Circuit Knitting: Circuit cutting

QuasiProbability Decomposition (QPD): $\mathcal{E}(\rho) = \sum_i a_i \mathcal{E}_i(\rho)$

Unitary channel (goal)

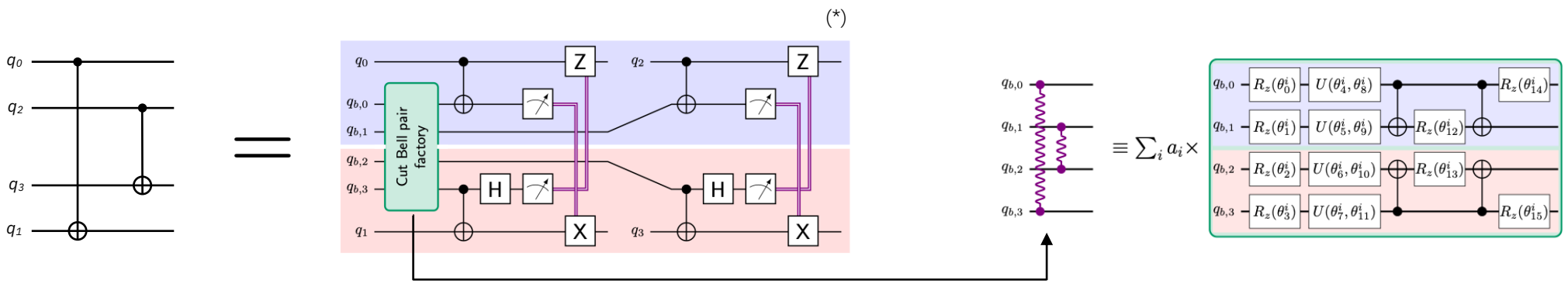
Coefficients (can be negative)

Unitary channels (available)

Convert to probability by defining $\gamma = \sum_i |a_i|$ and then $\mathcal{E}(\rho) = \gamma \sum_i p_i \mathrm{sign}(a_i) \mathcal{E}_i(\rho)$ with $p_i = |a_i|/\gamma$

This allows us to cut *gates* with execution overhead: γ^2 for one cut and γ^{2n} for n cuts

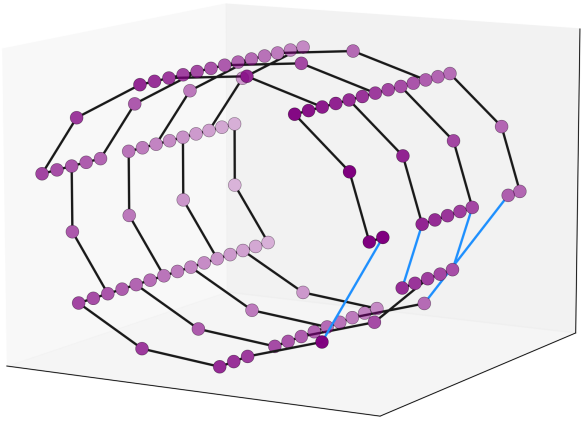
Example:



(*)Figure from Carrera Vázquez *et al. arXiv:2402.17833 (2024)*

Circuit Knitting: Circuit cutting

Application: periodic boundary conditions in a system with limited physical connectivity



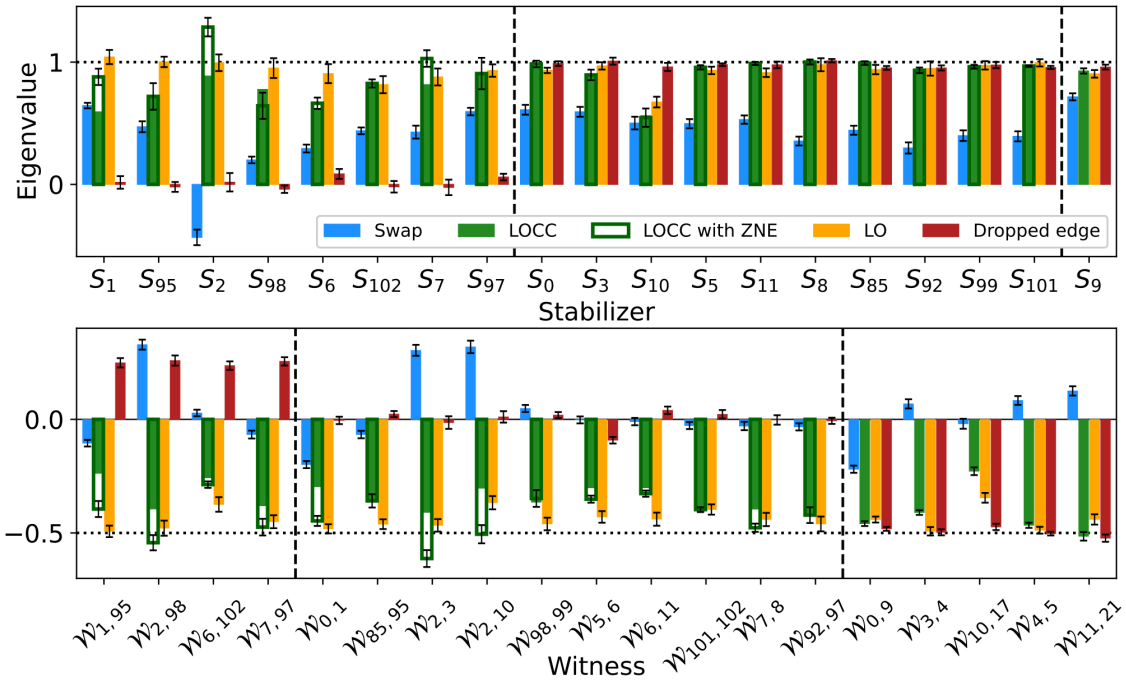
Eagle device (*ibm_kyiv*)

physical connection

virtual connection

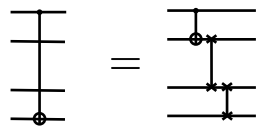
$$\mathcal{W}_{i,j} = (1 - \langle S_i \rangle - \langle S_j \rangle - \langle S_i S_j \rangle)/4$$

Node stabilizer — Edge stabilizer

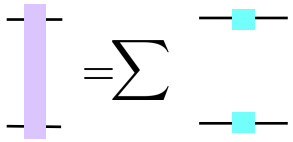


Three experiments:

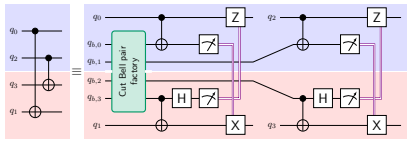
SWAP



LO



LOCC



Carrera Vázquez *et al. arXiv:*2402.17833 (2024)

Circuit Knitting: Entanglement forging

Schmidt decomposition of a pure $N+N$ qubit bipartite state: $|\Psi\rangle = (U \otimes V) \sum_{n=1}^{2^N} \underbrace{\lambda_n}_{\text{Schmidt coefficients}} |b_n\rangle \otimes |b_n\rangle$

Density matrix formalism:

$$|\Psi\rangle\langle\Psi| = (U \otimes V) \sum_{n=1}^{2^N} \left(\lambda_n^2 |b_n\rangle\langle b_n|^{\otimes 2} + \sum_{m=1}^{n-1} \lambda_n \lambda_m \sum_{p \in \mathbb{Z}_4} (-1)^p |\, \phi_{b_n b_m}^p \rangle\rangle\langle\langle \phi_{b_n b_m}^p \,|^{\otimes 2} \right) (U^\dagger \otimes V^\dagger)$$

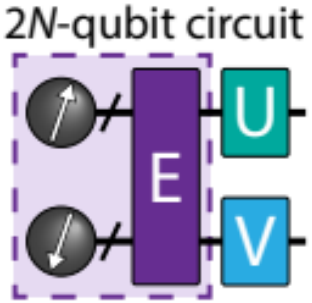
(with $|\phi_{xy}^p\rangle = \frac{(|x\rangle + i^p |y\rangle)}{\sqrt{2}}$ and $p \in \mathbb{Z}_4$)

Operator expectation value:

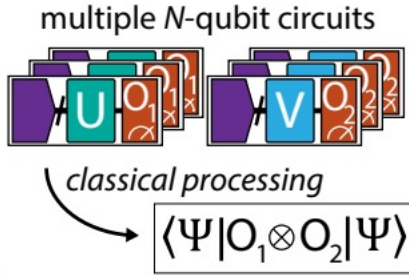
$$\langle O \rangle = \sum_{n=1}^{2^N} \left(\lambda_n^2 \langle b_n | \tilde{O}_1 | b_n \rangle \langle b_n | \tilde{O}_2 | b_n \rangle + \sum_{m=1}^{n-1} \lambda_n \lambda_m \sum_{p \in \mathbb{Z}_4} (-1)^p \left\langle \phi_{b_n b_m}^p \middle| \tilde{O}_1 \middle| \phi_{b_n b_m}^p \right\rangle \left\langle \phi_{b_n b_m}^p \middle| \tilde{O}_2 \middle| \phi_{b_n b_m}^p \right\rangle \right)$$

Circuit Knitting: Entanglement forging

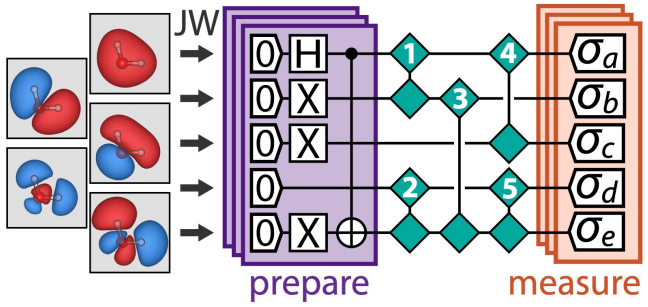
Original circuit to measure O:



With entanglement forging:

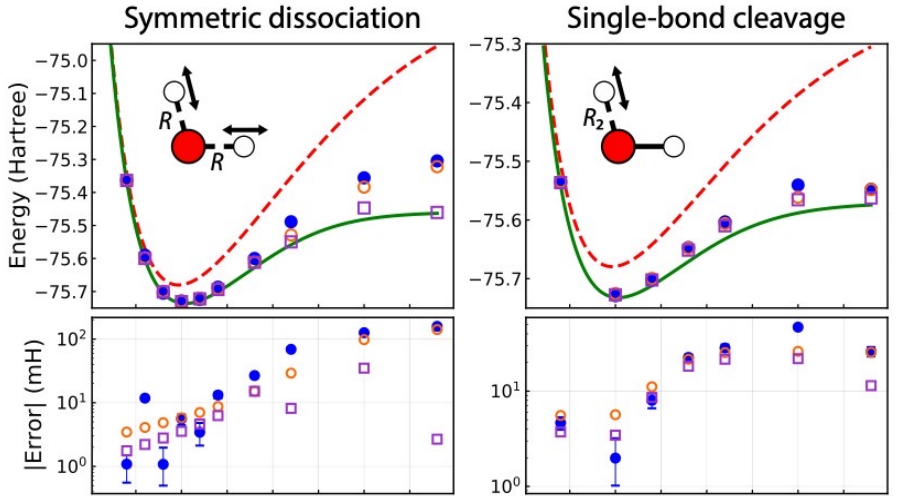


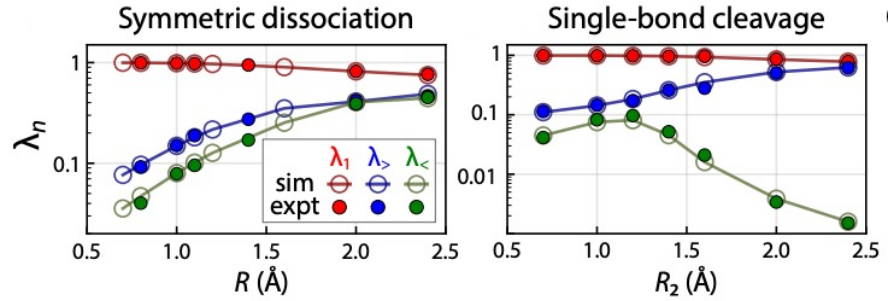
Example: H_20 molecule



Sampling overhead price:

$$\gamma^2 \sim \left(\sum_i |\lambda_i|\right)^4$$



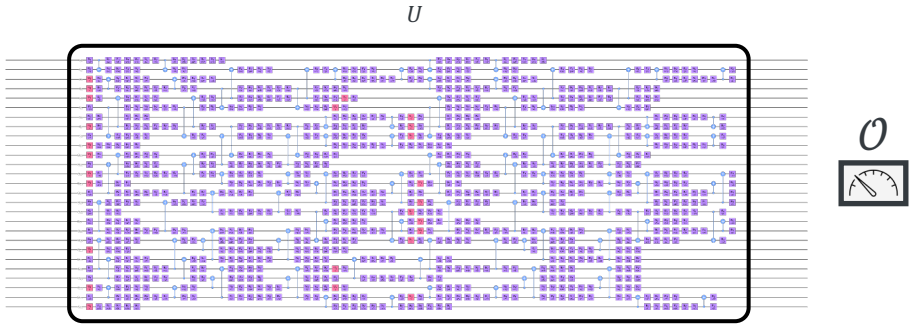


Eddins *et al. PRX Quantum* **3**, 010309 (2022)

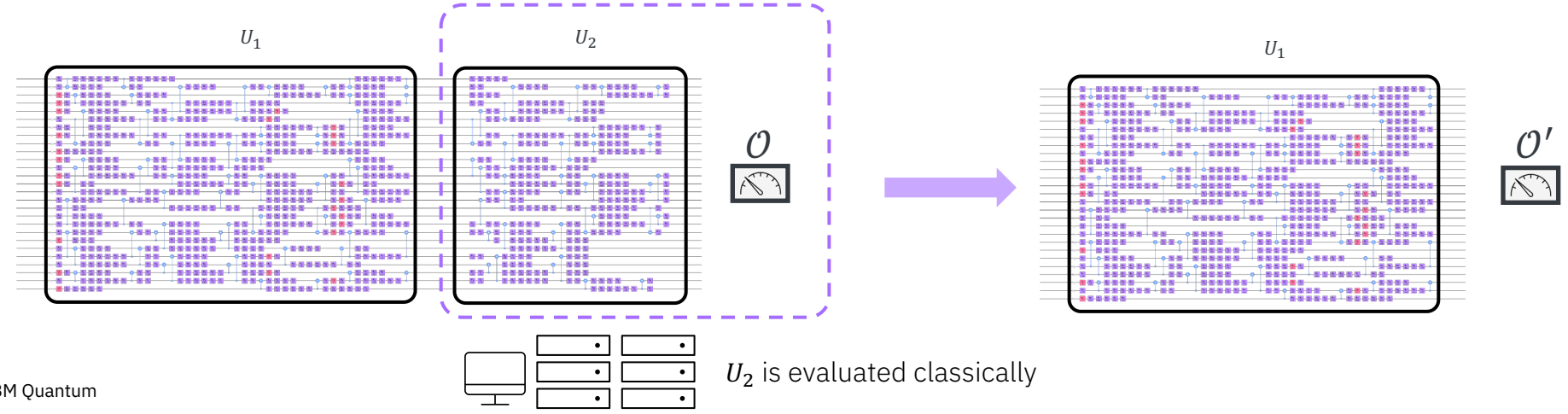
Circuit Knitting: OBP

Operator Backpropagation (OBP) *is a technique that uses HPC resources to trade quantum depth for an increased # of circuits and more difficult observables*

We want to estimate $\langle\psi|U^\dagger\mathcal{O}U|\psi\rangle$



We can express that value as $\langle\psi|U_1^\dagger U_2^\dagger\mathcal{O}U_2U_1|\psi\rangle$ and approximate instead $\langle\psi|U_1^\dagger\mathcal{O}'U_1|\psi\rangle$ with $U_2^\dagger\mathcal{O}U_2 \approx \mathcal{O}' = \sum_i c_i P_i$



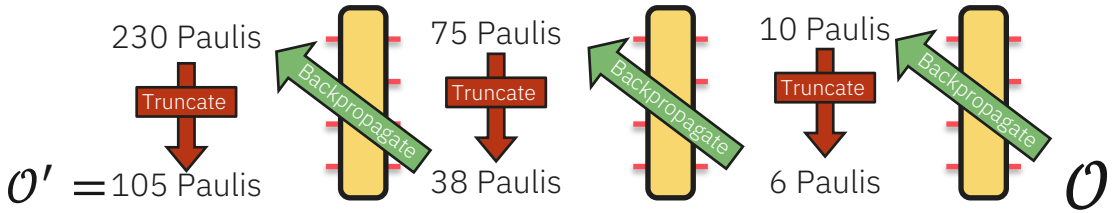
Circuit Knitting: OBP

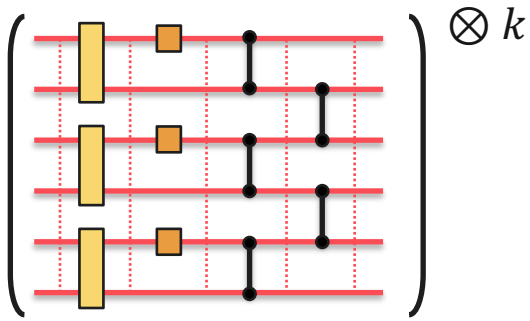
Operator Backpropagation (OBP) *is a technique that uses HPC resources to trade quantum depth for an increased # of circuits and more difficult observables*

OBP in practice

Clifford Perturbation Theory:

- Hardness scales with non-Cliffordness
- Accuracy-efficiency trade-off through truncation





4 slices

Can split error budget across slices

- Equal distribution for each slide ($\epsilon/4k$)
- Clifford gates use no budget, so we could do: $(\frac{\epsilon}{4k}, \frac{3\epsilon}{4k}, 0, 0)$'
- Evenly distributing the error would look like: $(\frac{\epsilon}{2k}, \frac{\epsilon}{2k}, 0, 0)$'

Circuit Knitting: Shor's algorithm

Problem: Given a composite integer $N \leq 2^n$ find one of its non-trivial divisors

Intuition:

Consider $N = pq$ and its multiplicative group modulo N, G

How many elements does G have?

Useful property: $x^{|G|} = 1 \bmod N$ Then, from $x^{p-1} = 1 \bmod p$

and its generalization $x^{(p-1)(q-1)} = 1 \bmod pq$, we get $|G| = (p-1)(q-1)$

Example: $N = 15$ Then $G = \{1,2,4,7,8,11,13,14\}$

And we get $|G| = 8 = (p-1)(q-1)$ from which, with $N = pq$, we obtain $p = 5$ and $q = 3$

Circuit Knitting: Shor's algorithm

Problem: Given a composite integer $N \leq 2^n$ find one of its non-trivial divisors

Solution:

Shor's algorithm does not produce $|G|$ but the period s of the function

$f(r) = x^r \bmod N$

This period is not $|G|$ but it *divides* $|G|$

Shor's algorithm: pick a random x relatively prime to N and get s

Then $x^s - 1 = 0 \bmod N \Rightarrow \left(x^{\frac{s}{2}} - 1\right)\left(x^{\frac{s}{2}} + 1\right) = 0 \bmod N$

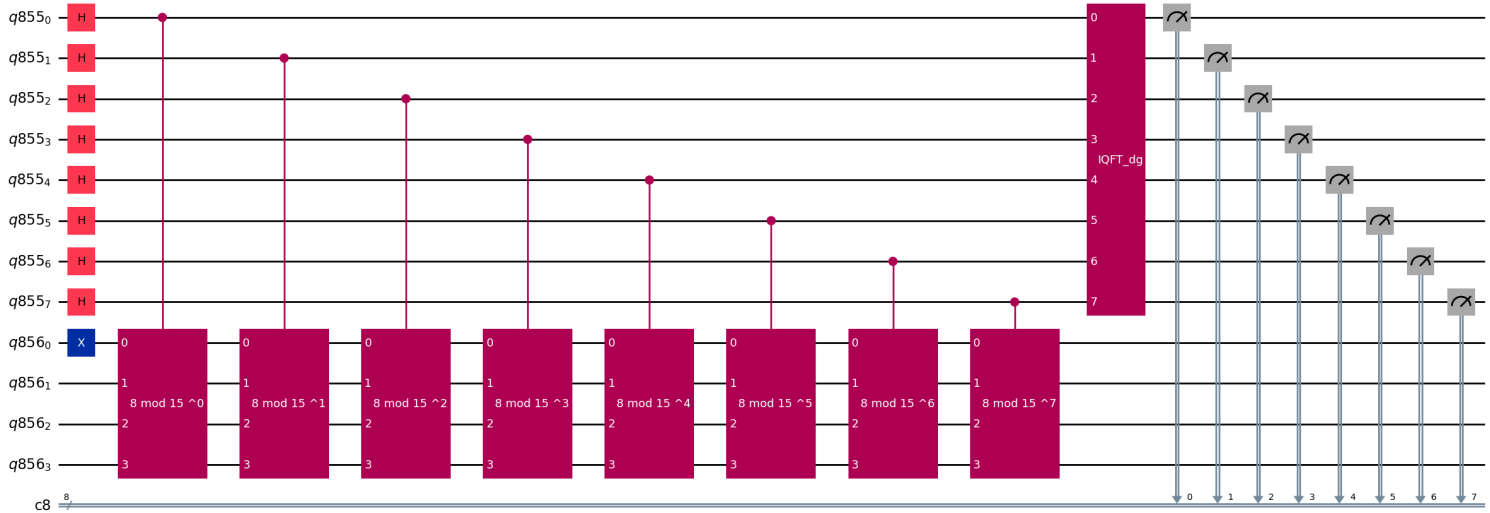
(if s is even)

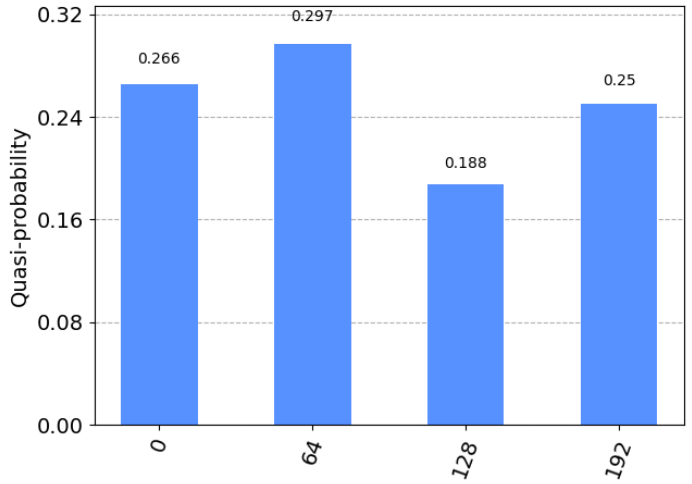
If neither $x^{s/2} - 1$ nor $x^{s/2} + 1$ are multiples of N, we get a non-trivial divisor of N by computing $\gcd(x^{\frac{s}{2}} - 1, N)$

Circuit Knitting: Shor's

Example: $N = 15$

Order-finding circuit





Output	Output/2^{2n}	r/s	Guess for s
0	0/256	0/1	1
64	64/256	1/4	4
128	128/256	1/2	2
192	192/256	3/4	4

and $2^{4/2} - 1 = 3$ and $\gcd(3,15) = 3$

Circuit Knitting: Regev's factoring algorithm[*]

Shor's complexity: $O(n^2 \log n)$ gates and $O(n)$ qubits[**]

Regev's algorithm uses two key ideas to improve on Shor's algorithm:

1) Work in higher dimensional space and replace x by several integers $y_1, \dots, y_d$

$$\mathcal{L} = \{(z_1, \dots, z_d) \in \mathbb{Z}^d \Big| \left(\prod_i {w_i}^{z_i} \right)^2 = 1 \, mod \, N\}$$

$$= \{(z_1, \dots, z_d) \in \mathbb{Z}^d \Big| \prod_i {y_i}^{z_i} = 1 \, mod \, N\} \subset \mathbb{Z}^d$$

2) Choose $y_1, \dots, y_d$ to be *very small* compared to N

This allows to perform modular exponentiation with only $\sim O(\frac{n^2}{d})$ operations, which results in $O(n^{3/2} \log n)$ gates

[*] Oded Regev, *arXiv*: 2308.06572 (2023)

[**] See Gidney and Ekerå, *Quantum* **5**, 433 (2021)

Circuit Knitting: Regev's

Shor's complexity: $O(n^2 \log n)$ gates and $O(n)$ qubits

Regev's algorithm requires $O(\sqrt{n})$ calls to the quantum circuit (vs $O(1)$ from Shor's) and $O(n^{3/2} \log n)$ gates

Two main limitations to Regev's algorithm:

1) It requires $O(n^{\frac{3}{2}})$ qubits, many more than Shor's

   Addressed by Ragavan and Vaikuntanathan[*], which recovered $O(n \log n)$ qubits

2) Lacks theoretical guarantees

   Addressed by Pilatte[**]

Next developments:

- Reduce memory further
- Resource estimation for $n = 2048$

[*] Ragavan and Vaikuntanathan, *arXiv*: 2310.00899 (2023)

[**]Cédric Pilatte, *arXiv*: 2404.16450 (2024)

Regev's result was extended to discrete logarithms by Ekerå and Gärtner: Ekera and Gartner, arXiv: 2311.05545 (2023)

Circuit Knitting: Multi-product Formulas

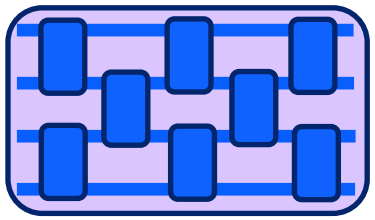
Multi-product Formulas (MPF) *is a technique that reduce the approximation error attained by product formulas for certain Hamiltonian simulation problems*

Brief review of Product Formulas

Solve $\frac{d}{dt}\rho(t) = -i[H, \rho(t)]$ ⟶ $\rho(t) = e^{-itH}\rho\, e^{itH}$

Assume $H = \sum_{a=1}^{d} F_a$ ⟶ $S(t) = e^{-itF_d} \ldots e^{-itF_2}e^{-itF_1} = e^{-itH} + O(t^2)$ (first order product formula)

Order p product formula if $S(t) = e^{-itH} + O(t^{p+1})$



⟶ For some interesting systems, it can be proven that[*] $\|\rho(t) - \rho_k(t)\|_1 \sim \frac{nt^3}{k^2}$ for second-order product formula (n is system size)

k Trotter steps: $\rho_k(t) = S(\frac{t}{k})^k \rho_{in} S(\frac{t}{k})^{-k}$

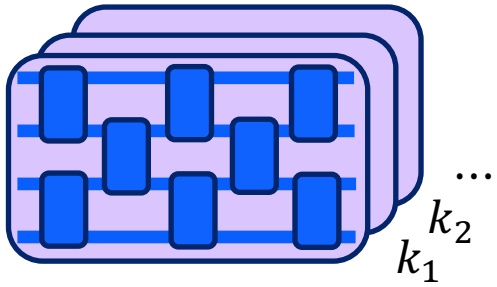
[*] Childs *et al. PRX* **11** 011020 (2021)

Circuit Knitting: Multi-product Formulas

Multi-product Formulas (MPF) *is a technique that reduce the approximation error attained by product formulas for certain Hamiltonian simulation problems*

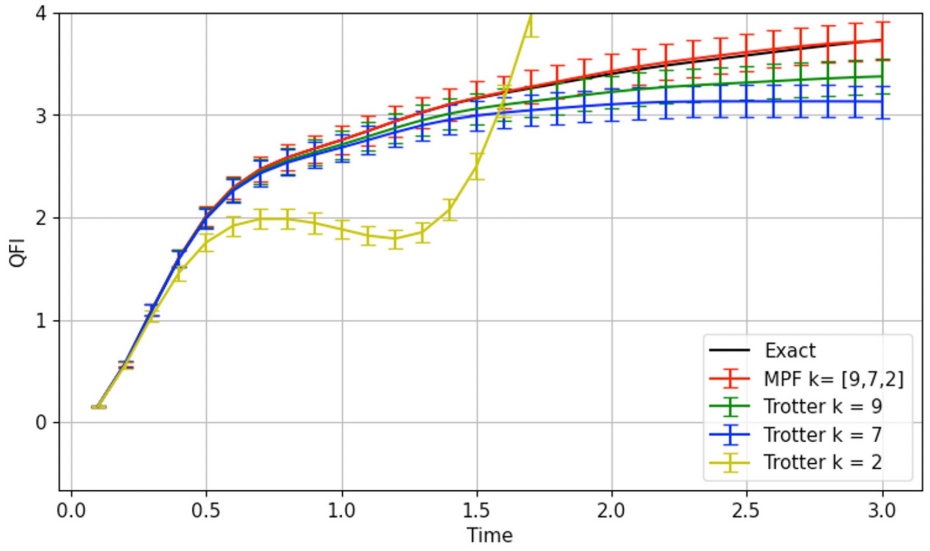
Multi-Product Formulas

Approximate the solution by a linear combination of product formulas $\mu(t) = \sum_{i=1}^{\chi} c_i \, \rho_{k_i}(t)$



Then it can be shown that for some interesting problems[*]

$$\|\rho(t) - \mu(t)\|_1 \sim n^2 t^6 \sum_{i=1}^{\chi} \frac{|c_i|}{k_i^4} = O(n^2 t^6 / k^4) \qquad (k = \max(k_1, k_2, \ldots))$$



Multi-product formulas can be understood as a form of Richardson extrapolation

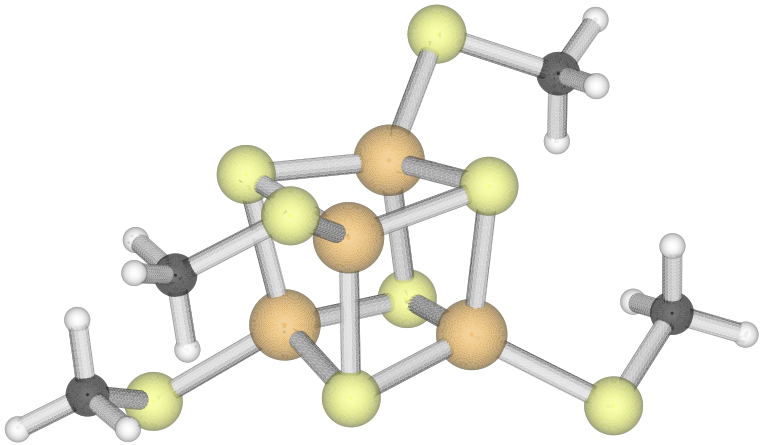
[*] Zhuk *et al. arXiv:* 2306.12569 (2024)

See also Carrera Vázquez *et al. Quantum* **7**, 1067 (2023)

Other QCSC Workflows: Sample-based Quantum Diagonalization (SQD)

The current belief is that pre-fault-tolerant quantum computers *in isolation* cannot deal with realistic formulations of nature

Other QCSC Workflows: Sample-based Quantum Diagonalization (SQD)



IBM Quantum RIKEN University of Colorado Boulder

Chemistry Beyond Exact Solutions on a Quantum-Centric Supercomputer

Javier Robledo-Moreno,[1,*] Mario Motta,[1,†] Holger Haas,[1] Ali Javadi-Abhari,[1] Petar Jurcevic,[1] William Kirby,[2] Simon Martiel,[3] Kunal Sharma,[1] Sandeep Sharma,[4] Tomonori Shirakawa,[5,6,7] Iskandar Sitdikov,[1] Rong-Yang Sun,[5,6,7] Kevin J. Sung,[1] Maika Takita,[1] Minh C. Tran,[2] Seiji Yunoki,[5,6,7,8] and Antonio Mezzacapo[1,‡]

[1] *IBM Quantum, IBM T.J. Watson Research Center, Yorktown Heights, NY 10598*
[2] *IBM Quantum, IBM Research Cambridge, Cambridge, MA 02142, USA*
[3] *IBM Quantum, IBM France Lab, Orsay, France*
[4] *Department of Chemistry, University of Colorado, Boulder, CO 80302, USA*
[5] *Computational Materials Science Research Team, RIKEN Center for Computational Science (R-CCS), Kobe, Hyogo, 650-0047, Japan*
[6] *Quantum Computational Science Research Team, RIKEN Center for Quantum Computing (RQC), Wako, Saitama, 351-0198, Japan*
[7] *RIKEN Interdisciplinary Theoretical and Mathematical Sciences Program (iTHEMS), Wako, Saitama 351-0198, Japan*
[8] *RIKEN Center for Emergent Matter Science (CEMS), Wako, Saitama 351-0198, Japan*

Robledo-Moreno *et al. arXiv*: 2405.05068 (2024)

Fe_4S_4 on **72** qubits (TZP-DKH basis set): **6.7M** Pauli operators

10^{-10} precision on each operator for chemical accuracy

Each circuit must be executed **10^{20}** times

Runtime at 10µs/circuit ~**3M years**

77 qubits
10570 quantum gates
3590 two-qubit gates

富岳
Fugaku

6400 nodes @
32 GB
1024 GB/s
48 cores

Other QCSC Workflows: Sample-based Quantum Diagonalization (SQD)

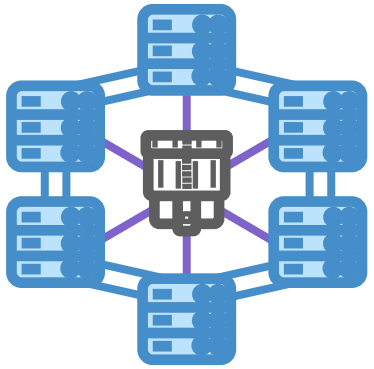
Intuition

We want to compute $\langle\psi|O|\psi\rangle$ for certain observable O with $|\psi\rangle = \sum_{n=0}^{2^{N-1}} \langle n|\psi\rangle\,|n\rangle$

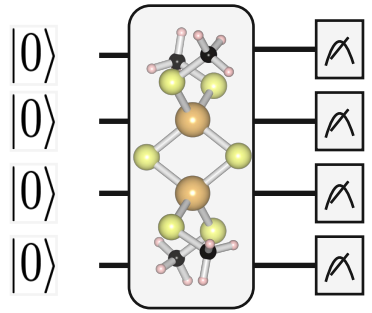
Then sampling on the computational basis, we can get the set of N_q bitstrings $S_{R=}\{|x\rangle\,|x \in \{0,1\}^{N_q}, R \text{ most frequent}\}$

And we solve $H_R\,c = E_R c\cdot$ in a classical computer, where $(H_R)_{xy} = \langle x|\hat{H}|y\rangle$ for $|x\rangle, |y\rangle \in S_R$

Components



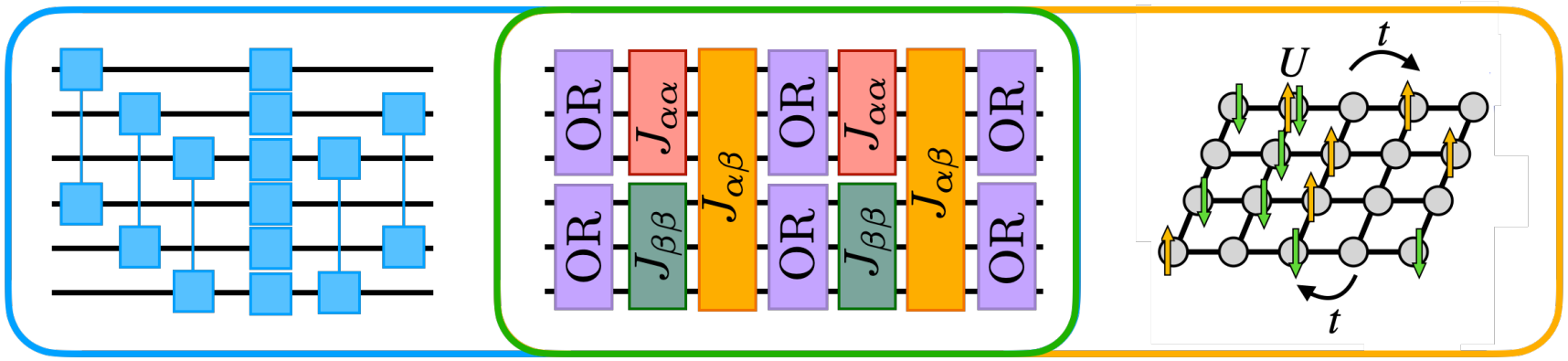
A quantum estimator that uses massive classical computing to process individual quantum samples



A class of quantum circuits of tunable depth for the accurate preparation of molecular ground states

Quantum circuits for chemistry: Local Unitary Coupled Jastrow

Hardware-efficient | **LUCJ circuit** | **Physically-motivated**



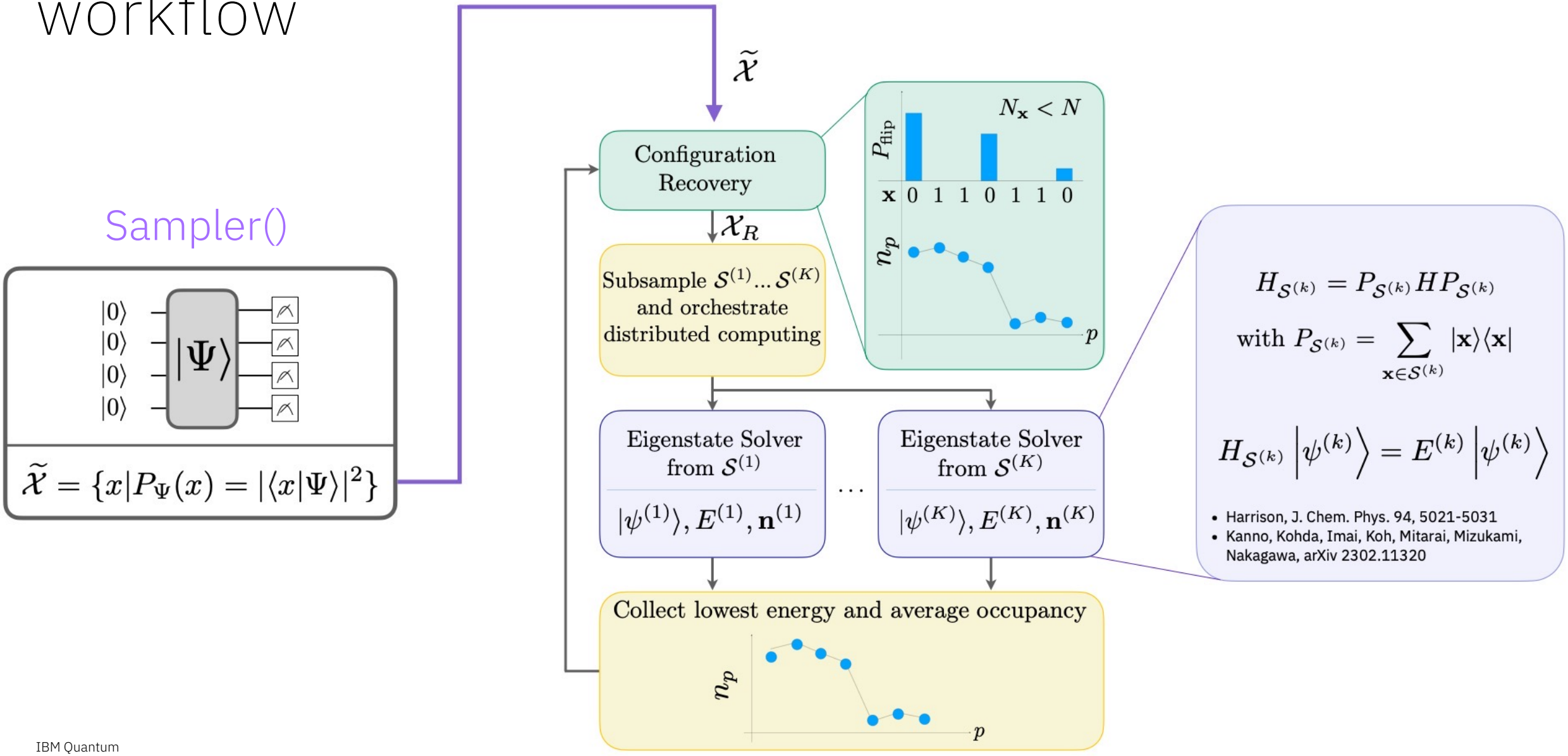
Compatible with hardware connectivity/gates/depth...
Hard to initialize and optimize

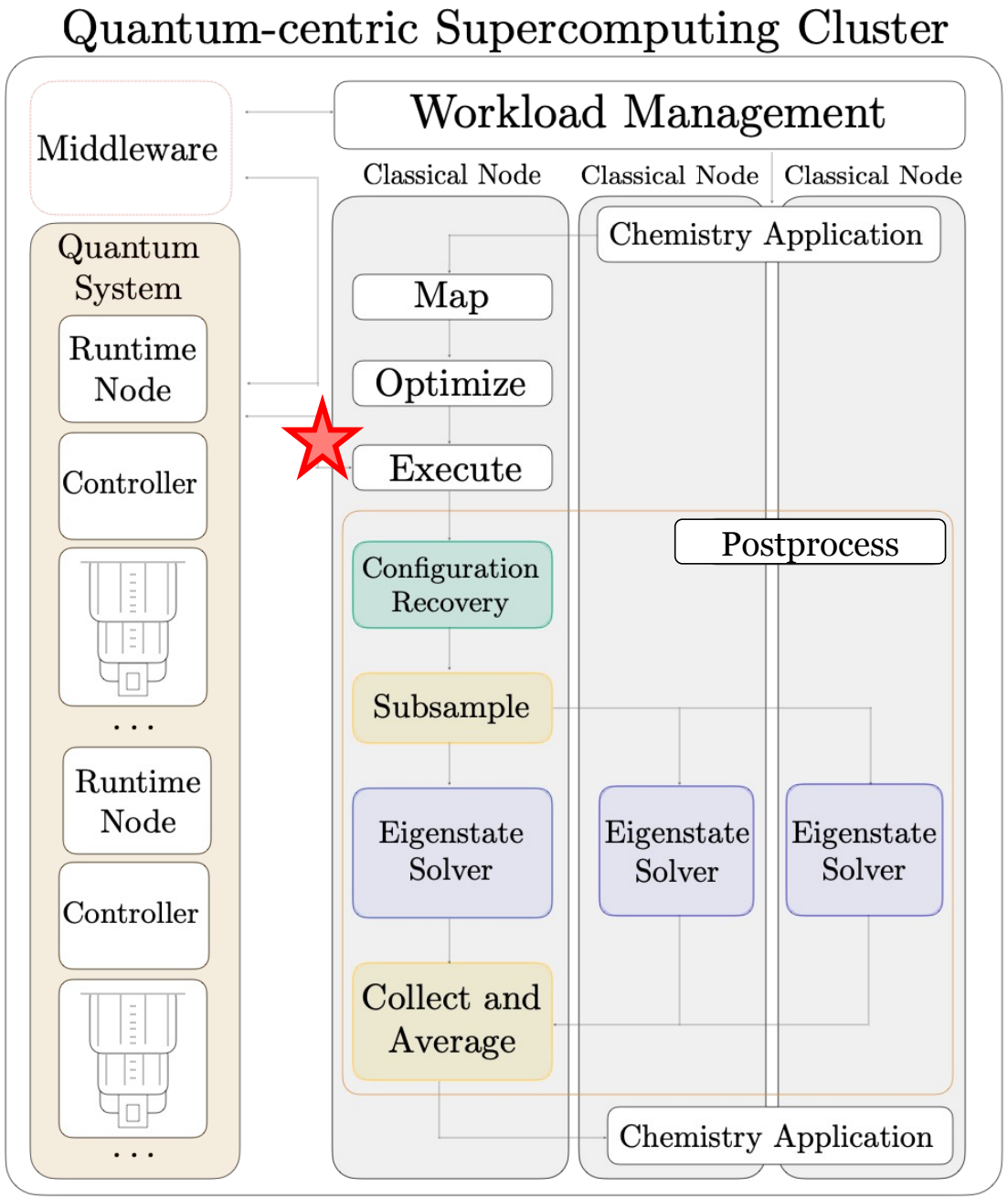
Parameters from classical calculations, easy to optimize
All-to-all connectivity, high circuit depth and gate count

Local Unitary Cluster Jastrow
Physical motivation (derived from coupled-cluster)
Hardware friendliness (device-specific connectivity)

HPC Quantum Estimator: Quantum-classical workflow

Sampler()





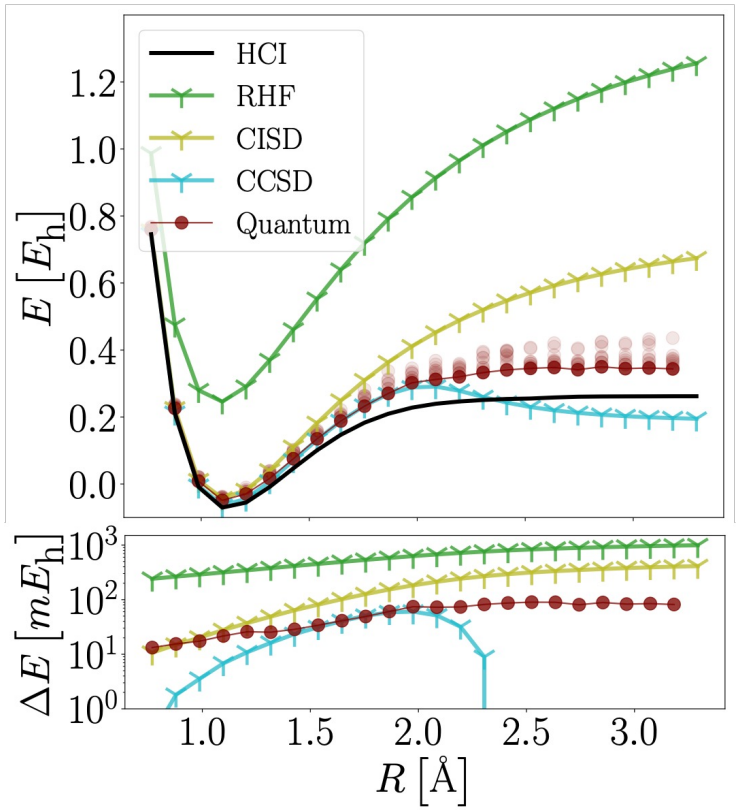
Typically, the data transfer between quantum and classical is limited to expectation values and circuit parameters

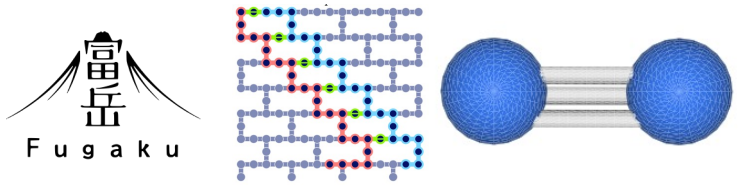
Here we use largest possible data transfer via the sampler primitive

Information from all the samples is used to perform accurate, noise-resistant estimations

Chemistry Beyond Exact Solutions on a Quantum-centric Supercomputer

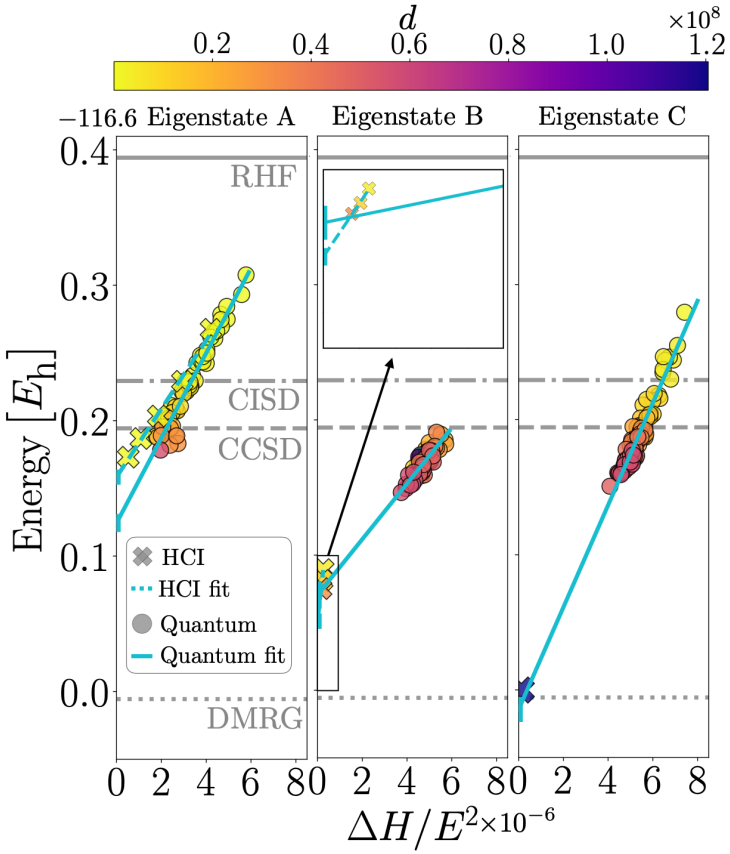
N_2 : Bond breaking on large basis set

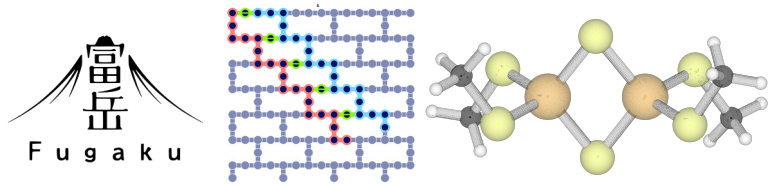




58 qubits
5204 quantum gates
1792 two-qubit gates

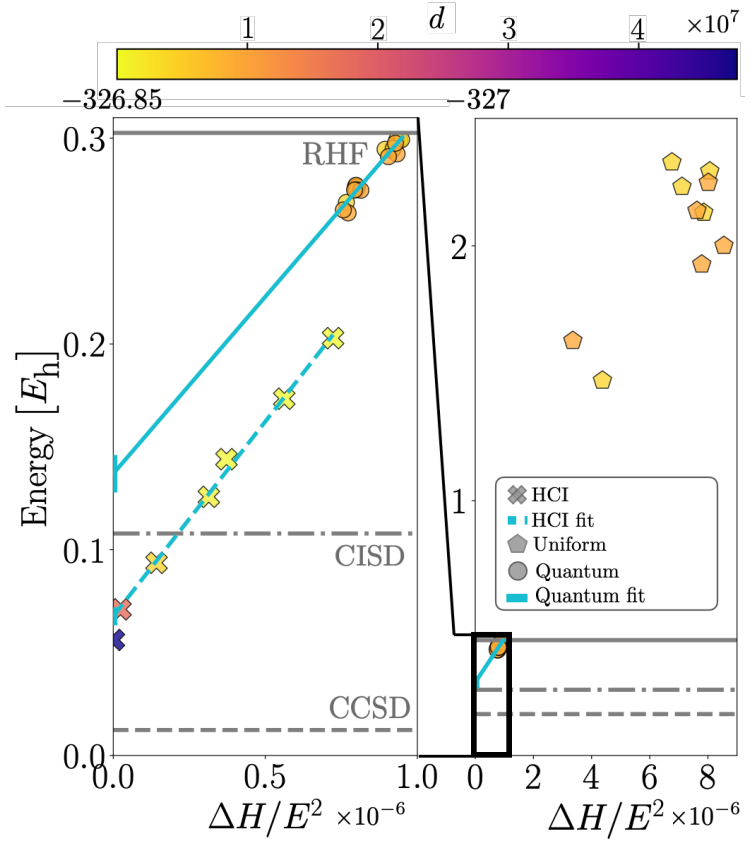
Fe_2S_2: Precision many-body physics

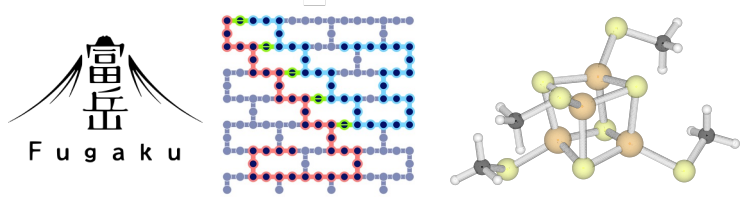




45 qubits
3170 quantum gates
1100 two-qubit gates

Fe_4S_4: Pushing hardware capabilities





77 qubits
10570 quantum gates
3590 two-qubit gates

IBM Quantum

Main Takeaways

Quantum-centric supercomputing enables realistic applications for our quantum processors, beyond problems tailored to the devices

---

Chemistry is a first example: classical distributed computing processes big classical data, while quantum executes a few large quantum circuits

---

Quantum simulations of molecules beyond the reach of exact classical solutions: 77 qubits, 3.5k two-qubit gates with a useful quantum signal

---

Processing of quantum data at the sample level: no false positive solutions and certifiable advantage

---

Increased classical processing extracts better quantum signals from circuits beyond exact simulations

IBM **Quantum**