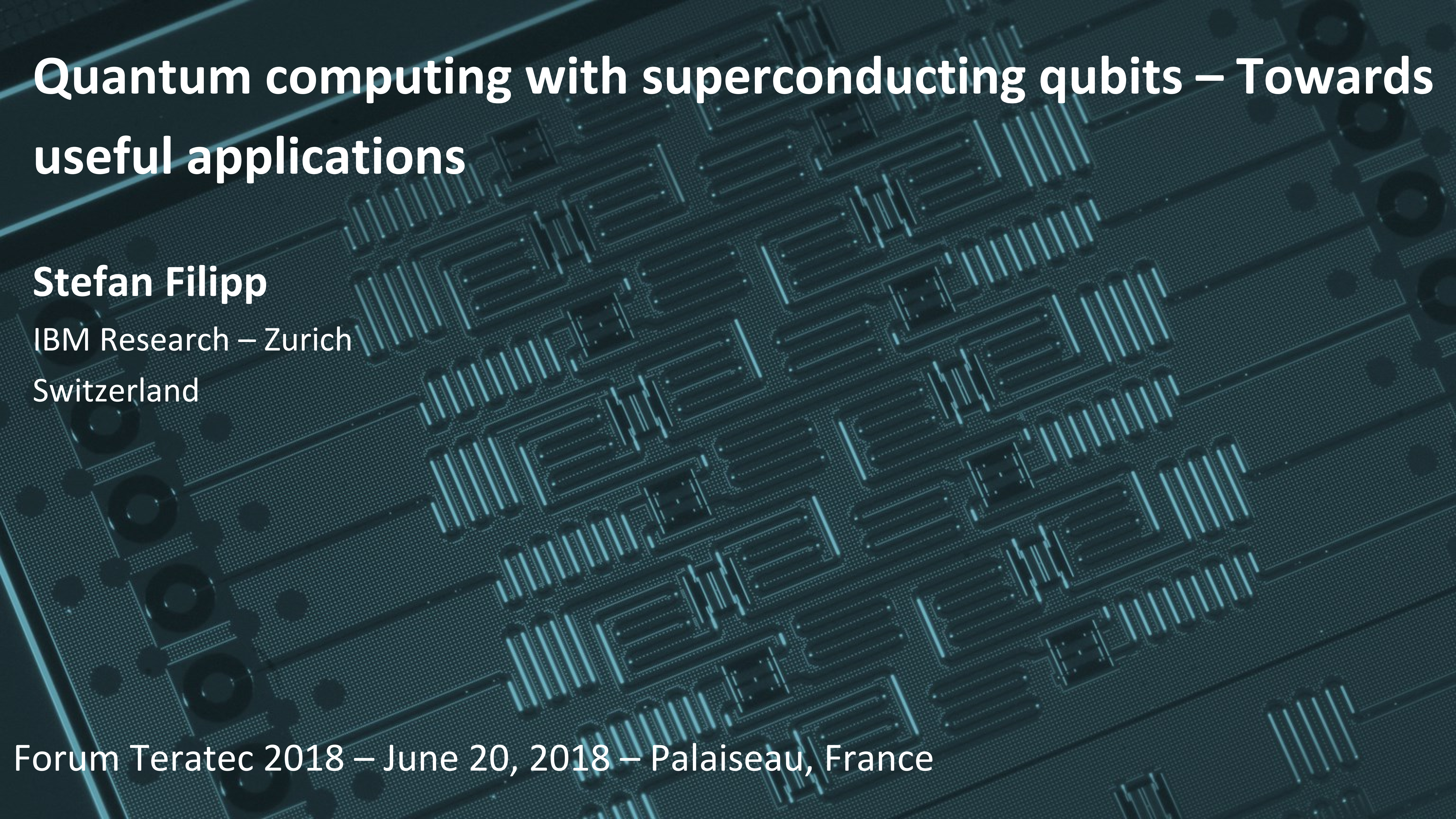


Quantum computing with superconducting qubits – Towards useful applications

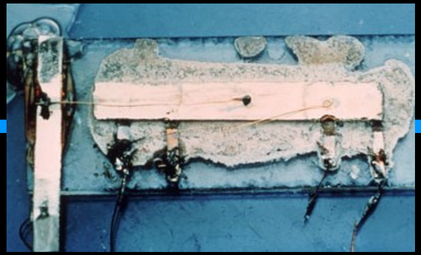
Stefan Filipp

IBM Research – Zurich
Switzerland

Forum Teratec 2018 – June 20, 2018 – Palaiseau, France

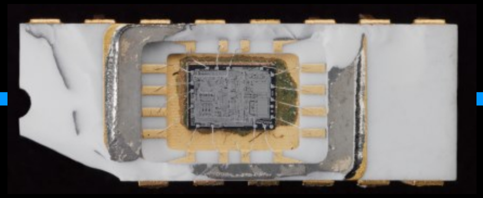
Why Quantum Computing? Why now?

1958



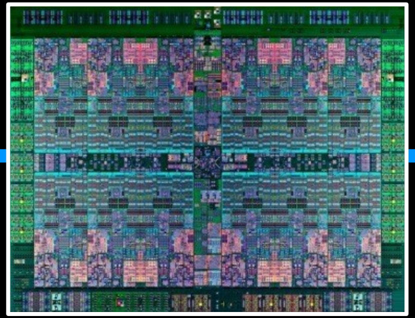
First integrated circuit
Size ~$1 cm^2$
2 Transistors

1971

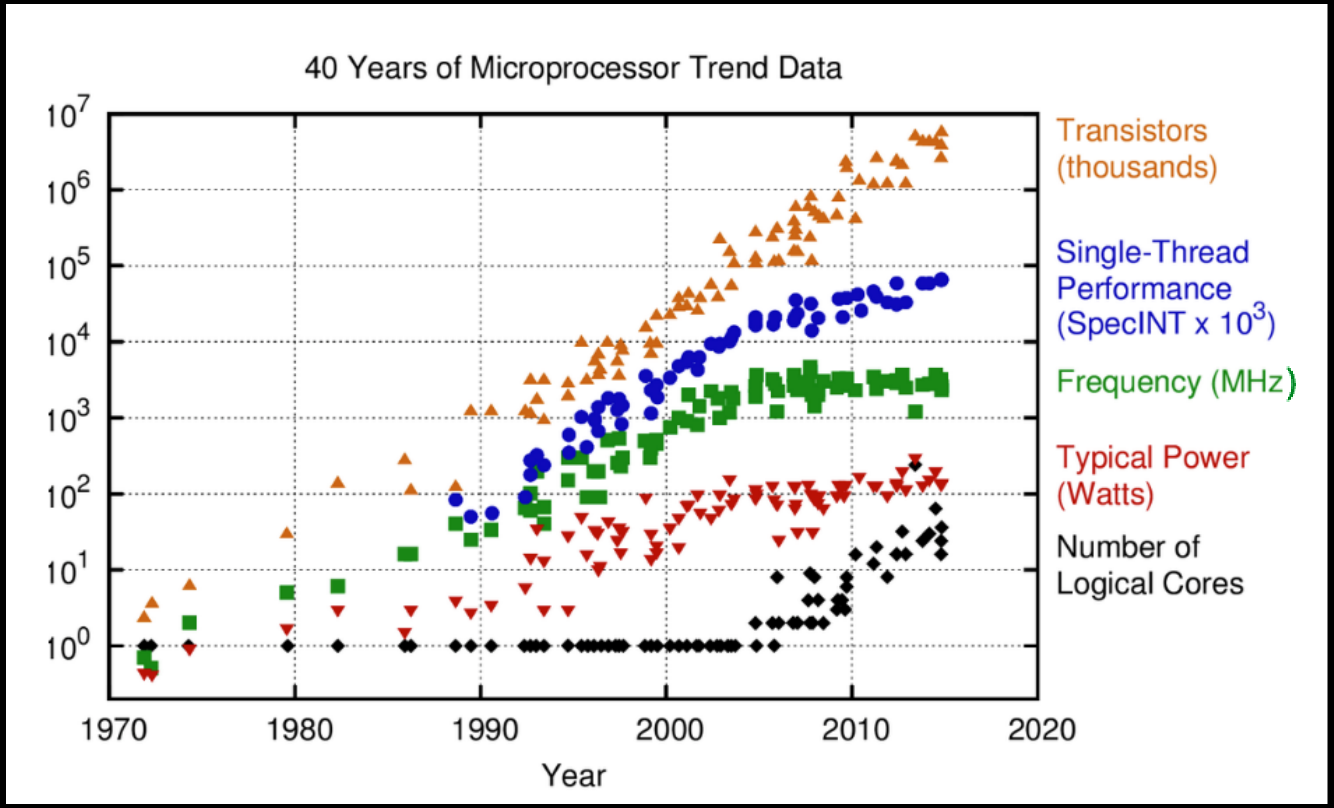


Moore's Law is Born
Intel 4004
2,300 transistors

2014

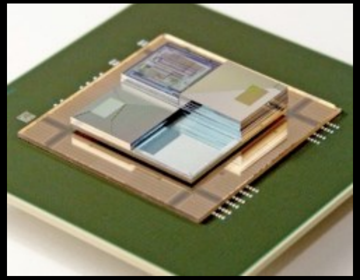


IBM P8 Processor ~ $650 mm^2$
22 nm feature size, 16 cores
> 4.2 Billion Transistors



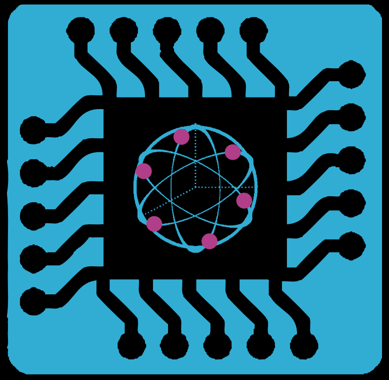
Alternative (co-existing) architectures:

next generation systems (3D/hybrid)



neuromorphic (cognitive)

quantum computing



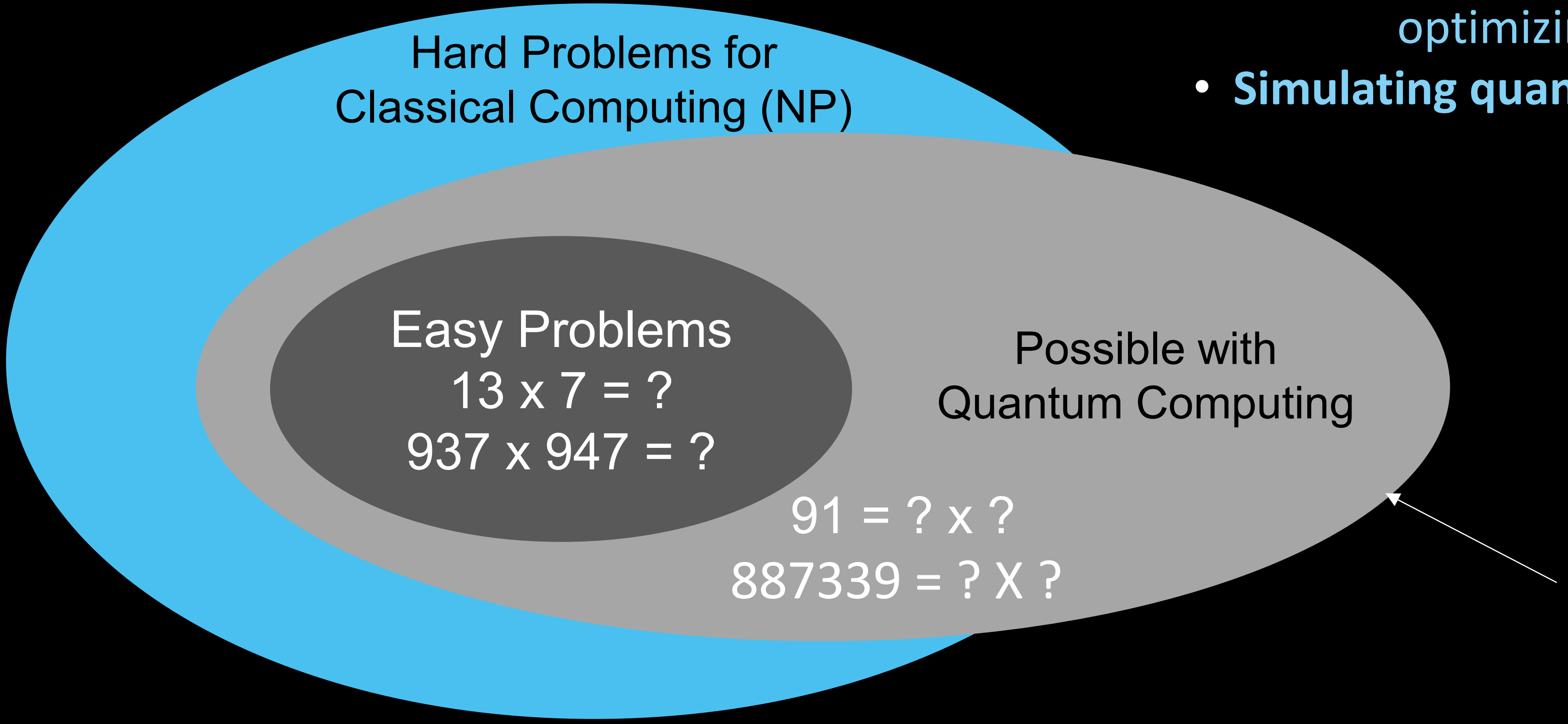
Quantum Computing as a path to solve intractable problems

Many problems in business and science are too complex for classical computing systems

“hard” / intractable problems:

(exponentially increasing resources with problem size)

- **Algebraic algorithms** (e.g. factoring, systems of equations) for machine learning, cryptography,…
- **Combinatorial optimization** (traveling salesman, optimizing business processes, risk analysis,…)
- **Simulating quantum mechanics** (chemistry, material science,…)



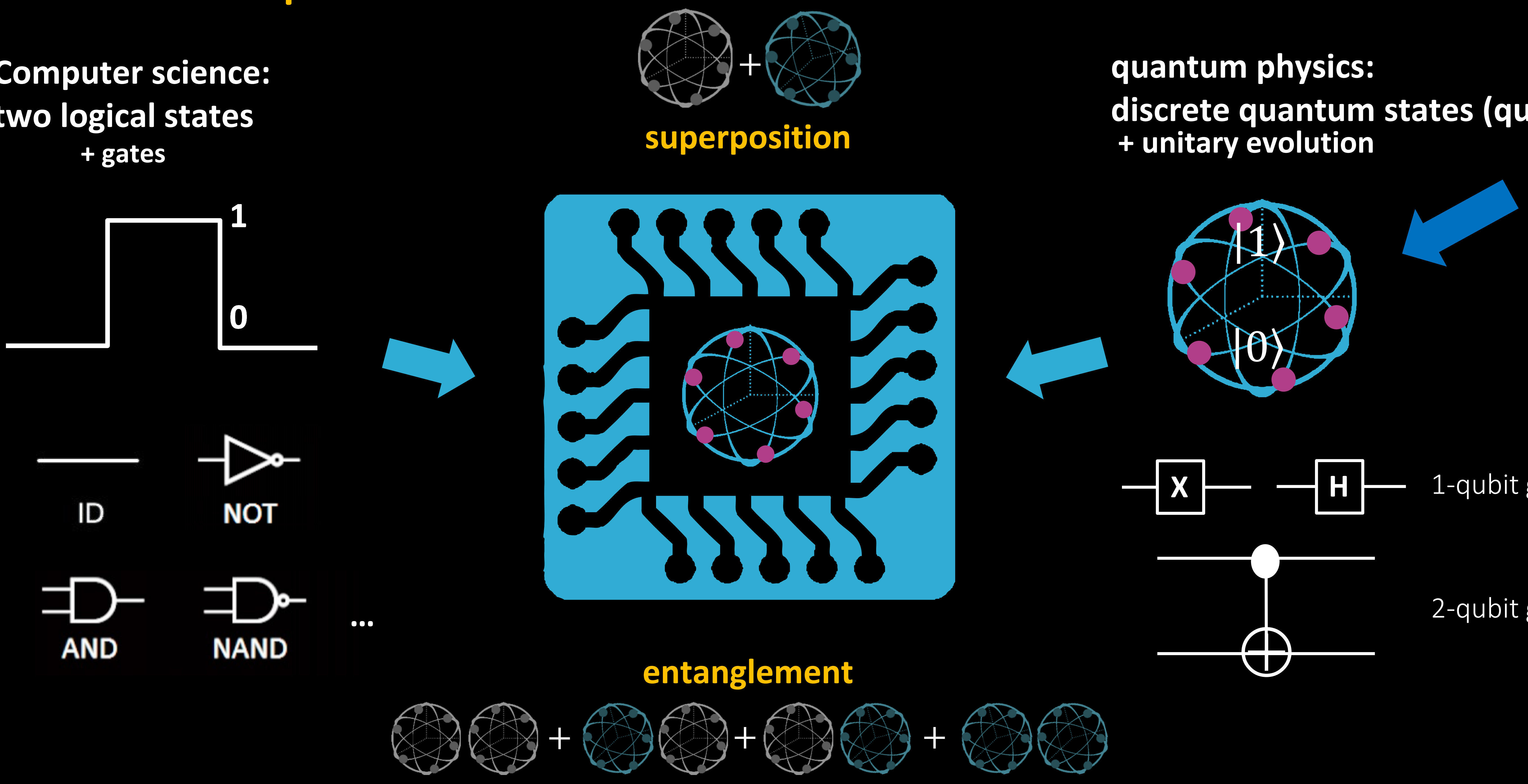
Material, Chemistry

Machine Learning

Optimization

Quantum computation

$$\alpha|0\rangle + \beta|1\rangle$$



superposition

Computer science:
two logical states
+ gates

quantum physics:
discrete quantum states (qubits)
+ unitary evolution

1-qubit gates

2-qubit gates

entanglement

$$\alpha|00\rangle + \beta|10\rangle + \gamma|01\rangle + \delta|11\rangle$$

The Quantum Advantage – Simulation of physical systems

How much memory is needed to store a quantum state?

How much time does it take to calculate dynamics of a quantum system?

# qubits	quantum state	coefficients	# bytes	timescale
1	$a\|0\rangle + b\|1\rangle$	$2^1 = 2$	16 Bytes	
2	$a\|00\rangle + b\|01\rangle + c\|10\rangle + d\|11\rangle$	$2^2 = 4$	32 Bytes	Nanoseconds
8		$2^8 = 256$	2kB	Microseconds on watch
16	…	$2^{16} = 65'536$	512 kB	Milliseconds on smartphone
32	…	~4 billion	32 GB	Seconds on laptop
64	…	~ information in internet	128 EB (134 million GB)	Years on supercomputer
256	…	~ # of atoms in universe	…	never

classical

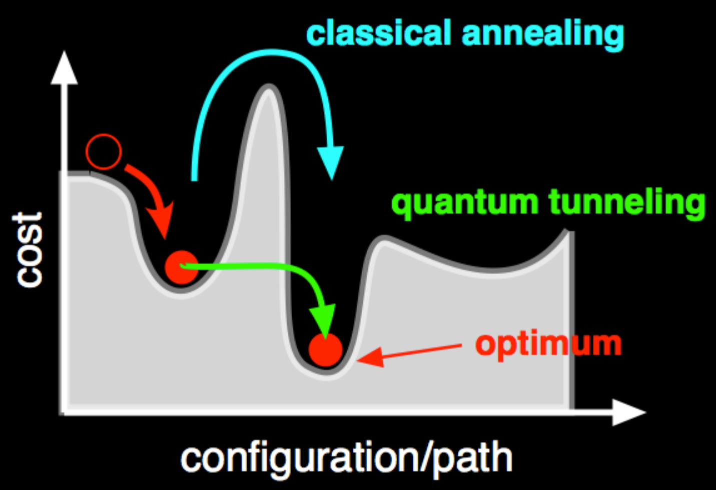
quantum

Types of Quantum Computing

Quantum Annealing

Optimization Problems

- Machine learning
- Fault analysis
- Resource optimization
- etc…

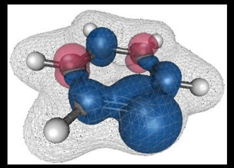


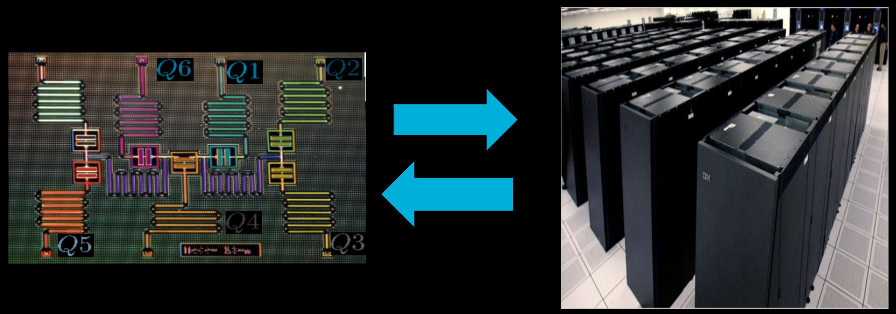
Many ‘noisy’ qubits can be built;
large problem class in optimization;
amount of quantum speedup unclear

Approximate Q-Comp.

Simulation of Quantum Systems, Optimization

- Material discovery
- Quantum chemistry
- Optimization
  (logistics, time scheduling,…)
- Machine Learning



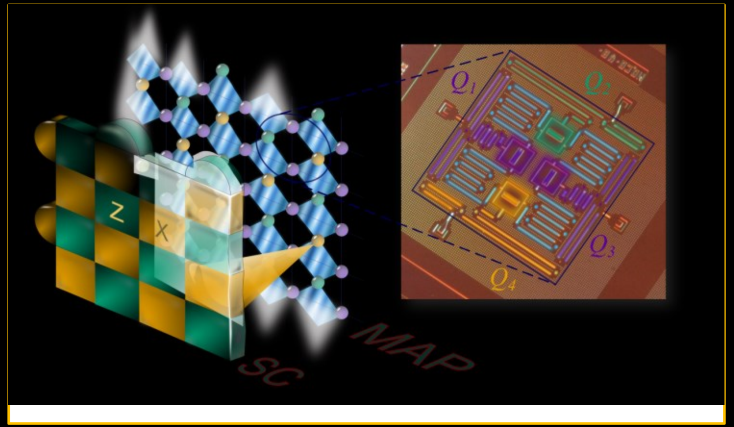


Hybrid quantum-classical approach;
already 50-100 “good” physical qubits
could provide quantum speedup.

Fault-tolerant Universal Q-Comp.

Execution of Arbitrary Quantum Algorithms

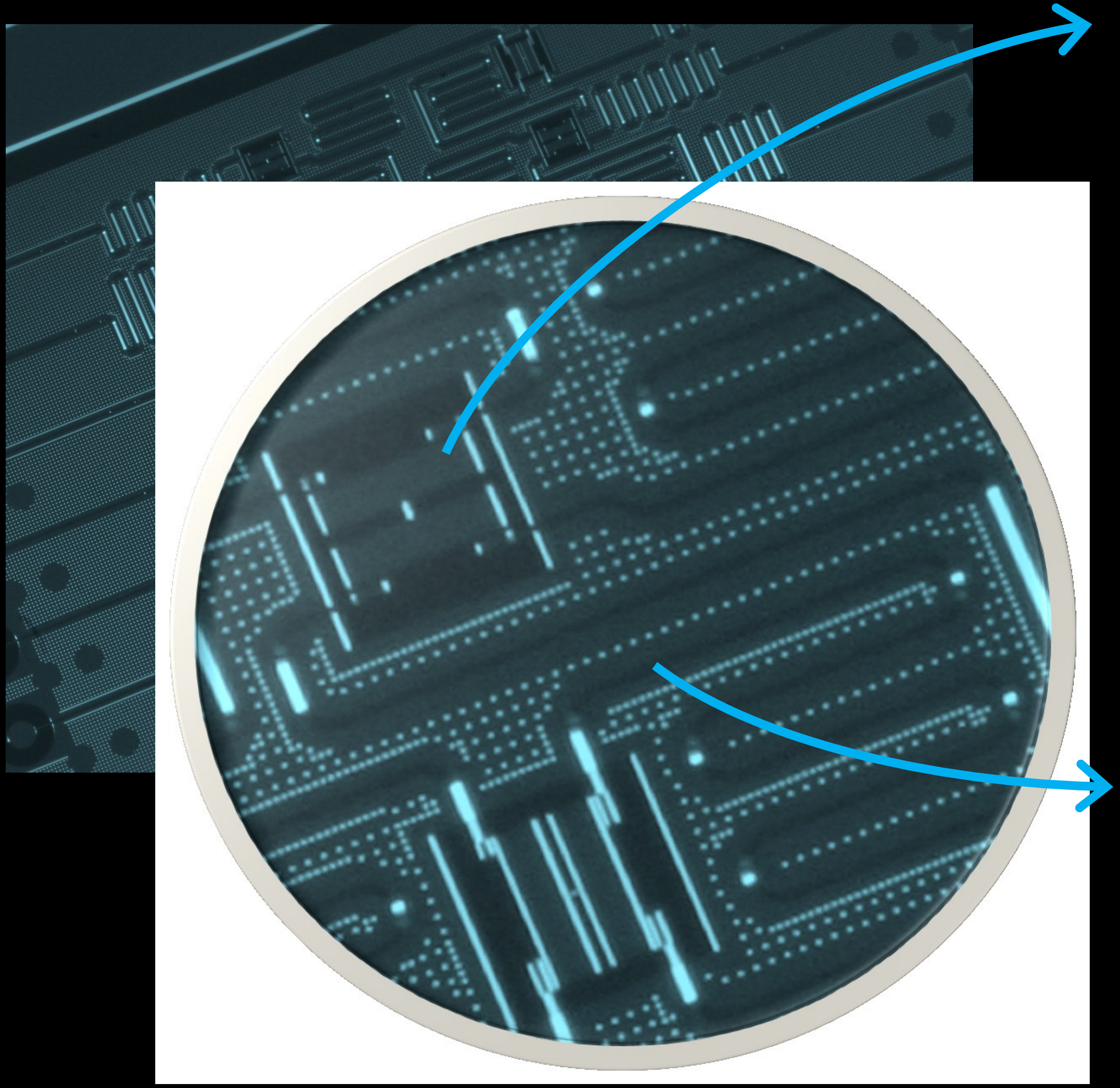
- Algebraic algorithms
  (machine learning, cryptography,…)
- Combinatorial optimization
- Digital simulation of quantum systems



Surface Code: Error correction in a Quantum Computer

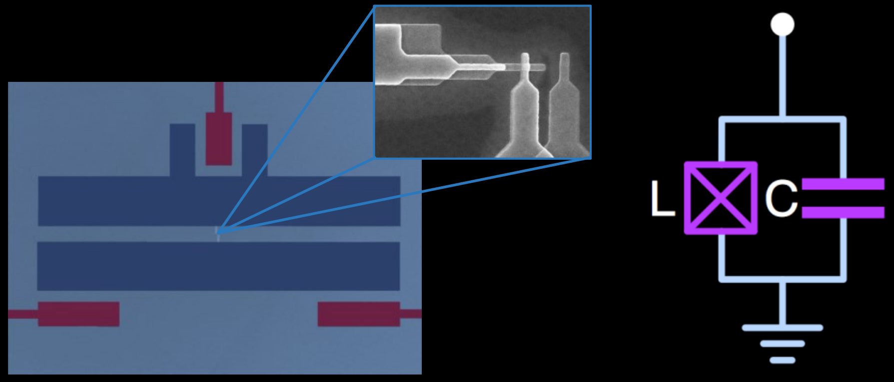
Proven quantum speedup;
error correction requires significant qubit overhead.

IBM: Superconducting Qubit Processor



Superconducting qubit

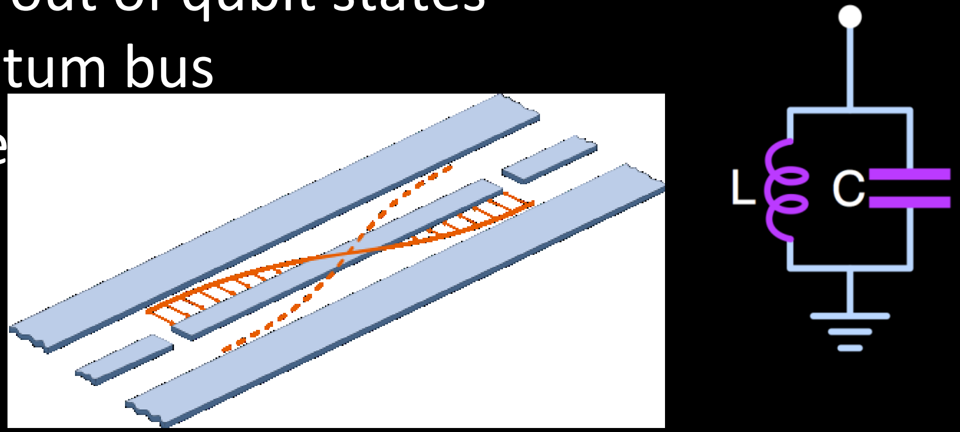
- quantum information carrier



$E_{01} \approx 5\ \text{GHz} \approx\ 240\ \text{mK}$

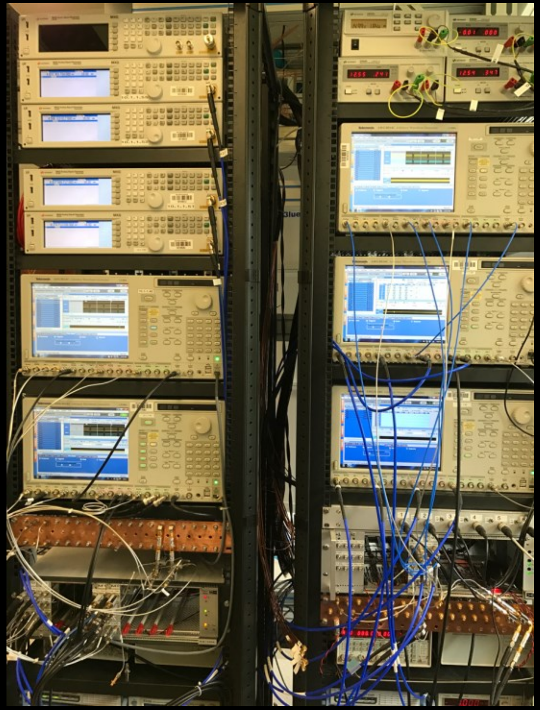
Microwave resonator:

- read-out of qubit states
- quantum bus
- noise



IBM Q quantum computing systems

Room Temperature



Microwave electronics

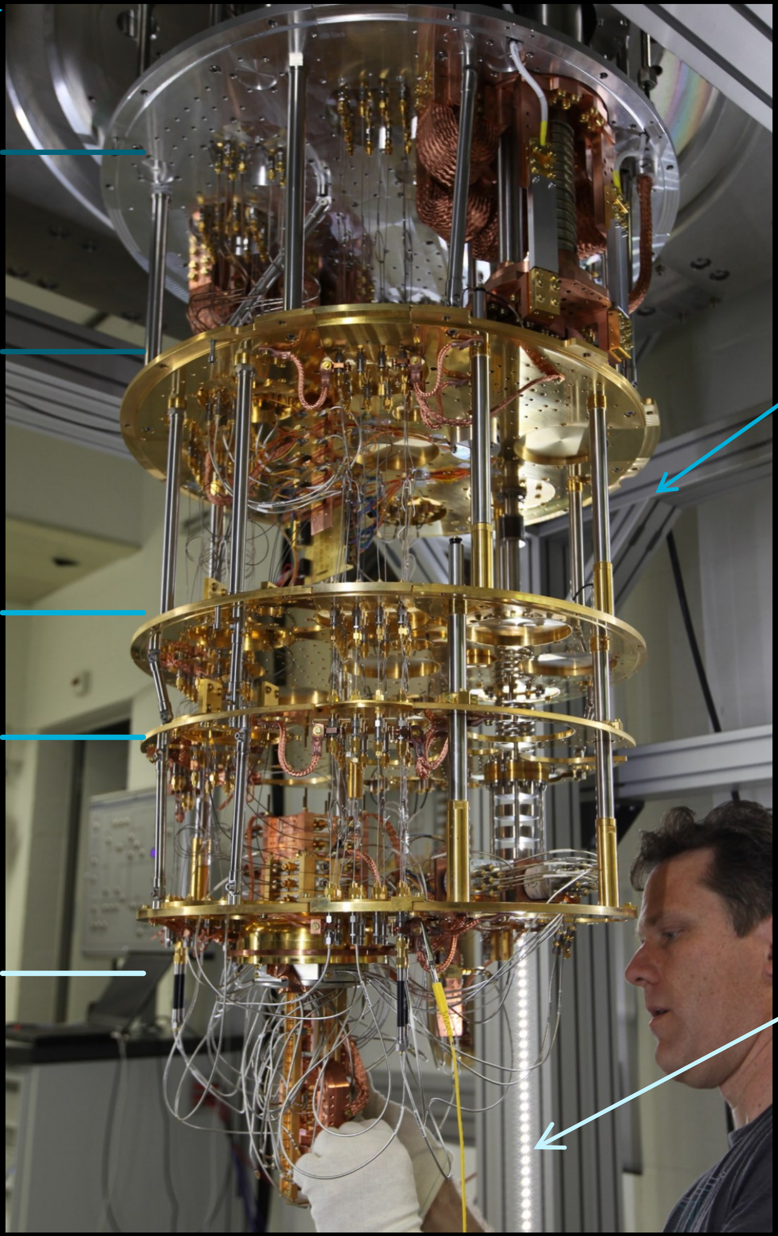
40K

3K

0.9K

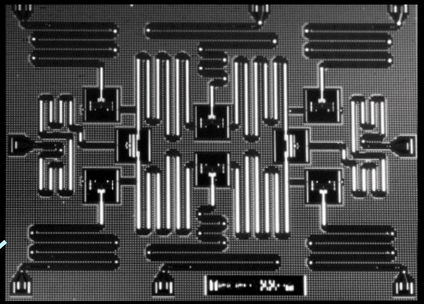
0.1K

0.015K

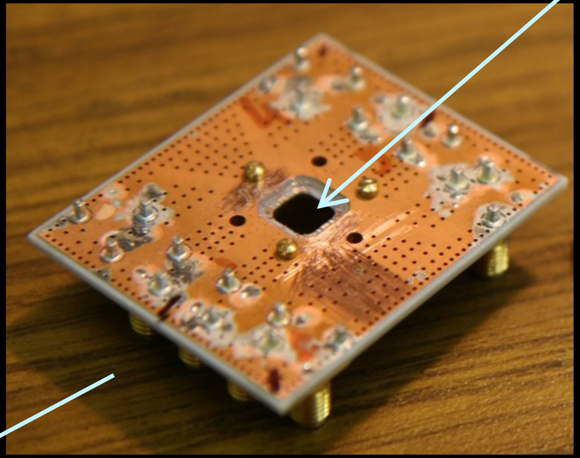


Refrigerator to cool qubits to 15 mK
with a mixture of ^{3}He and ^{4}He

Cosmic Microwave Background 2.7K



Chip with superconducting qubits and resonators

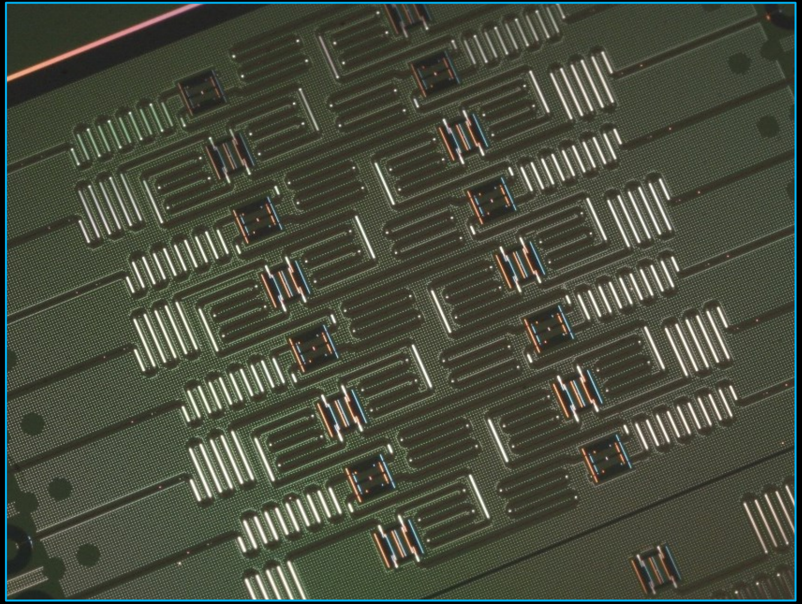


PCB with the qubit chip at 15 mK
Protected from the environment by multiple shields

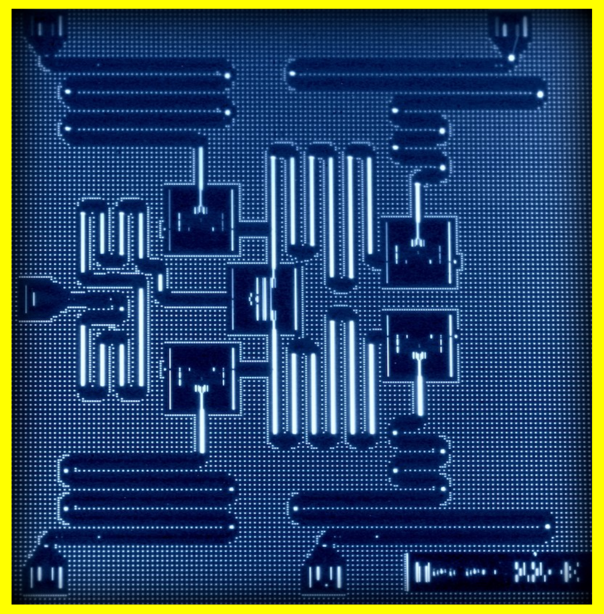
IBM qubit processor architectures

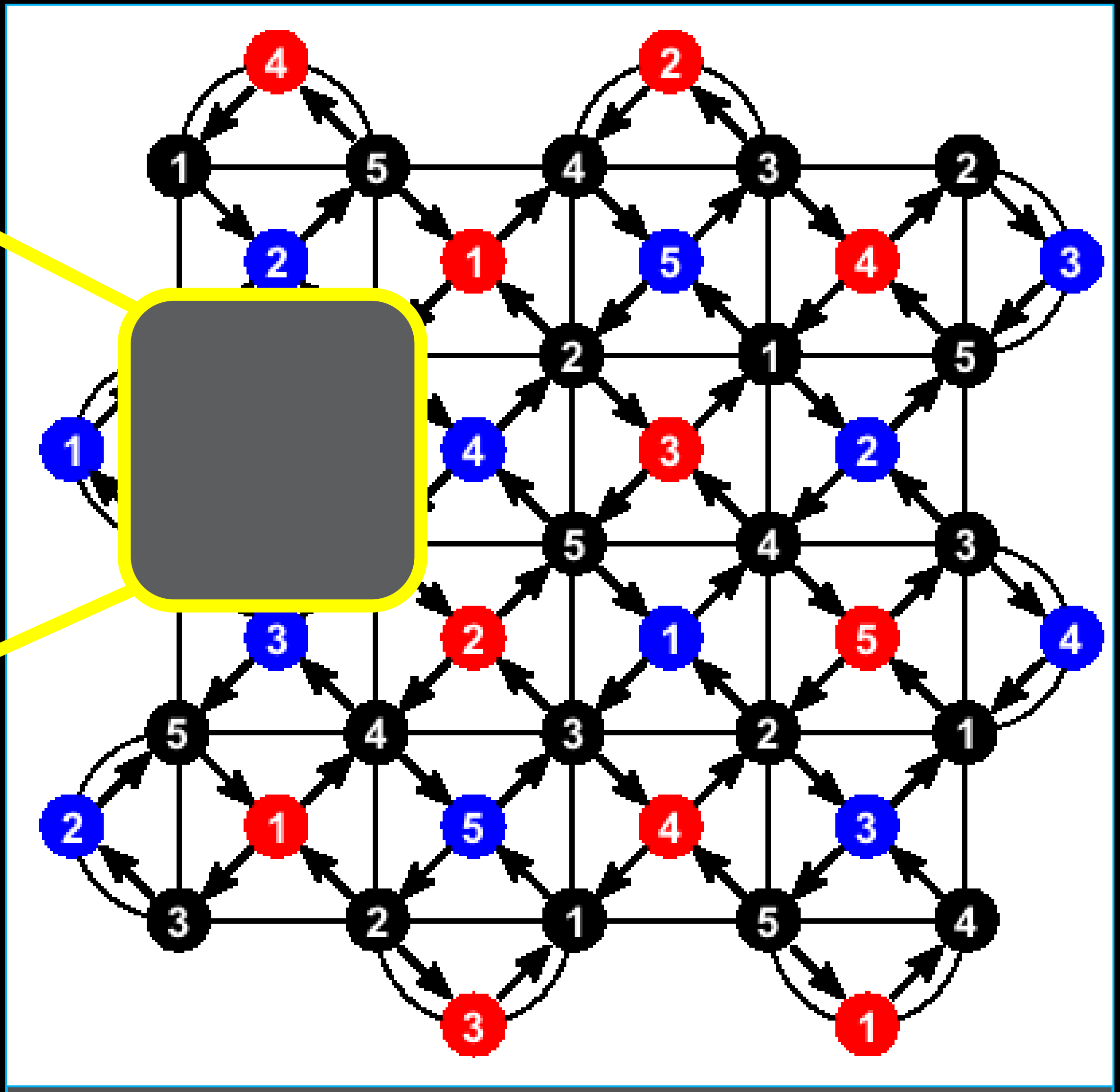
IBM Q experience (publicly accessible)

16 Qubits (2017)



5 Qubits (2016)

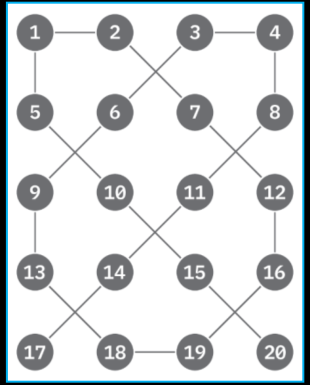




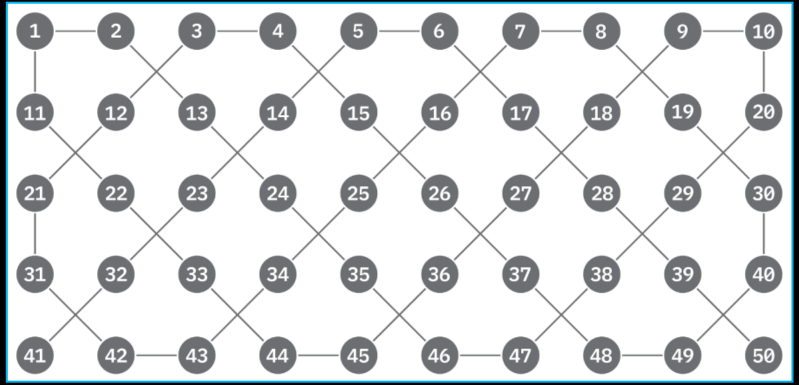
Latticed arrangement for scaling

IBM Q commercial

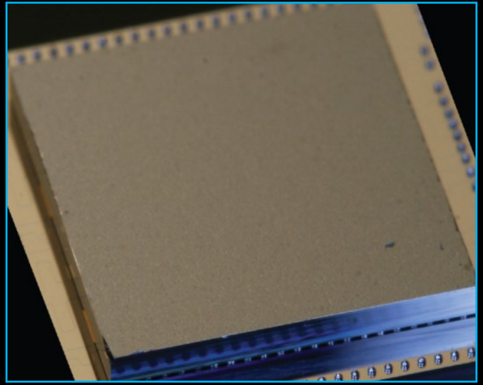
20 Qubits



50 Qubit architecture (2017)



Package

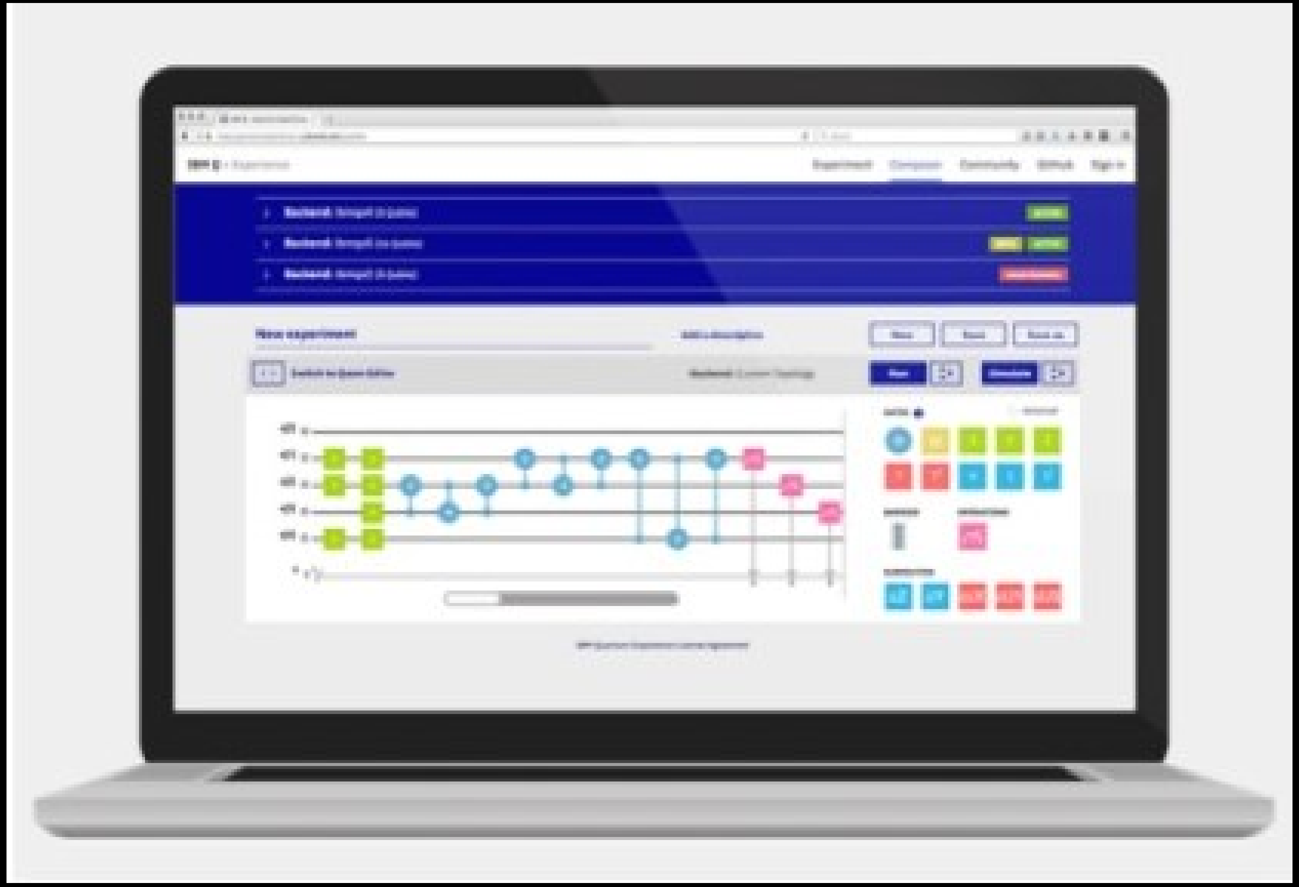


IBM Quantum Experience

Public quantum computer (up to 16 qubits) and developer ecosystem

IBM QX Features

- Tutorial
- Simulation
- Graphical programming
- QASM language
- API & SDK
- Active user community

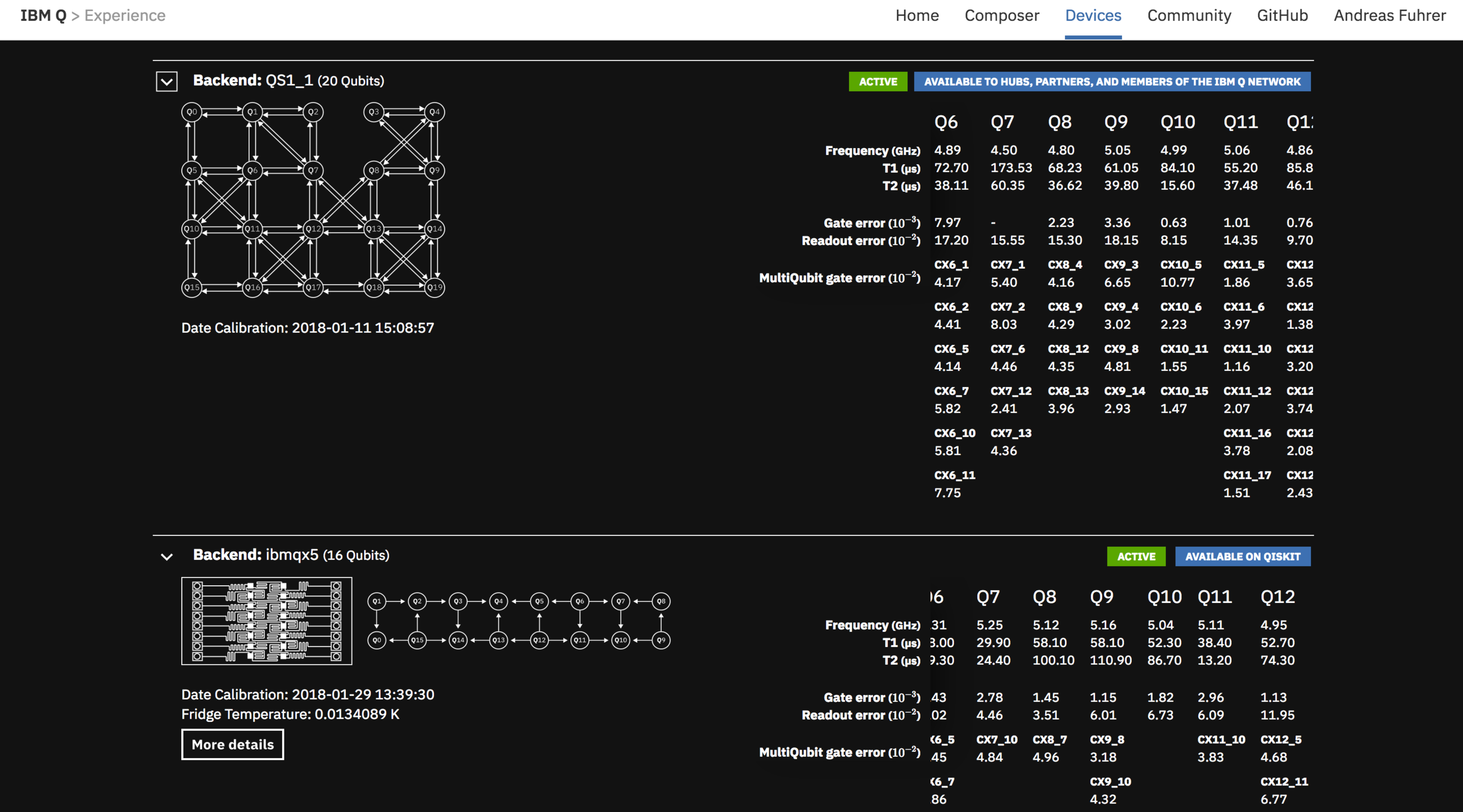


Since launch

- > 80,000 users
- > 3,000,000 experiments
- > 60 research papers
- used by 1,500+ colleges and universities, 300 high schools, 300 private institutions

Experience quantum computing here:

research.ibm.com/ibm-qx



Backend: QS1_1 (20 Qubits)

Date Calibration: 2018-01-11 15:08:57

	Q6	Q7	Q8	Q9	Q10	Q11	Q1
Frequency (GHz)	4.89	4.50	4.80	5.05	4.99	5.06	4.86
T1 (µs)	72.70	173.53	68.23	61.05	84.10	55.20	85.8
T2 (µs)	38.11	60.35	36.62	39.80	15.60	37.48	46.1
Gate error (10^{-3})	7.97	-	2.23	3.36	0.63	1.01	0.76
Readout error (10^{-2})	17.20	15.55	15.30	18.15	8.15	14.35	9.70
MultiQubit gate error (10^{-2})	**CX6_1** 4.17 **CX6_2** 4.41 **CX6_5** 4.14 **CX6_7** 5.82 **CX6_10** 5.81 **CX6_11** 7.75	**CX7_1** 5.40 **CX7_2** 8.03 **CX7_6** 4.46 **CX7_12** 2.41 **CX7_13** 4.36	**CX8_4** 4.16 **CX8_9** 4.29 **CX8_12** 4.35 **CX8_13** 3.96	**CX9_3** 6.65 **CX9_4** 3.02 **CX9_8** 4.81 **CX9_14** 2.93	**CX10_5** 10.77 **CX10_6** 2.23 **CX10_11** 1.55 **CX10_15** 1.47	**CX11_5** 1.86 **CX11_6** 3.97 **CX11_10** 1.16 **CX11_12** 2.07 **CX11_16** 3.78 **CX11_17** 1.51	**CX12** 3.65 **CX12** 1.38 **CX12** 3.20 **CX12** 3.74 **CX12** 2.08 **CX12** 2.43

Backend: ibmqx5 (16 Qubits)

Date Calibration: 2018-01-29 13:39:30
Fridge Temperature: 0.0134089 K

	)6	Q7	Q8	Q9	Q10	Q11	Q12
Frequency (GHz)	31	5.25	5.12	5.16	5.04	5.11	4.95
T1 (µs)	3.00	29.90	58.10	58.10	52.30	38.40	52.70
T2 (µs)	9.30	24.40	100.10	110.90	86.70	13.20	74.30
Gate error (10^{-3})	43	2.78	1.45	1.15	1.82	2.96	1.13
Readout error (10^{-2})	02	4.46	3.51	6.01	6.73	6.09	11.95
MultiQubit gate error (10^{-2})	**X6_5** 45 **X6_7** 86	**CX7_10** 4.84	**CX8_7** 4.96	**CX9_8** 3.18 **CX9_10** 4.32		**CX11_10** 3.83	**CX12_5** 4.68 **CX12_11** 6.77

Quantum Volume: How powerful is "my" Quantum Computer

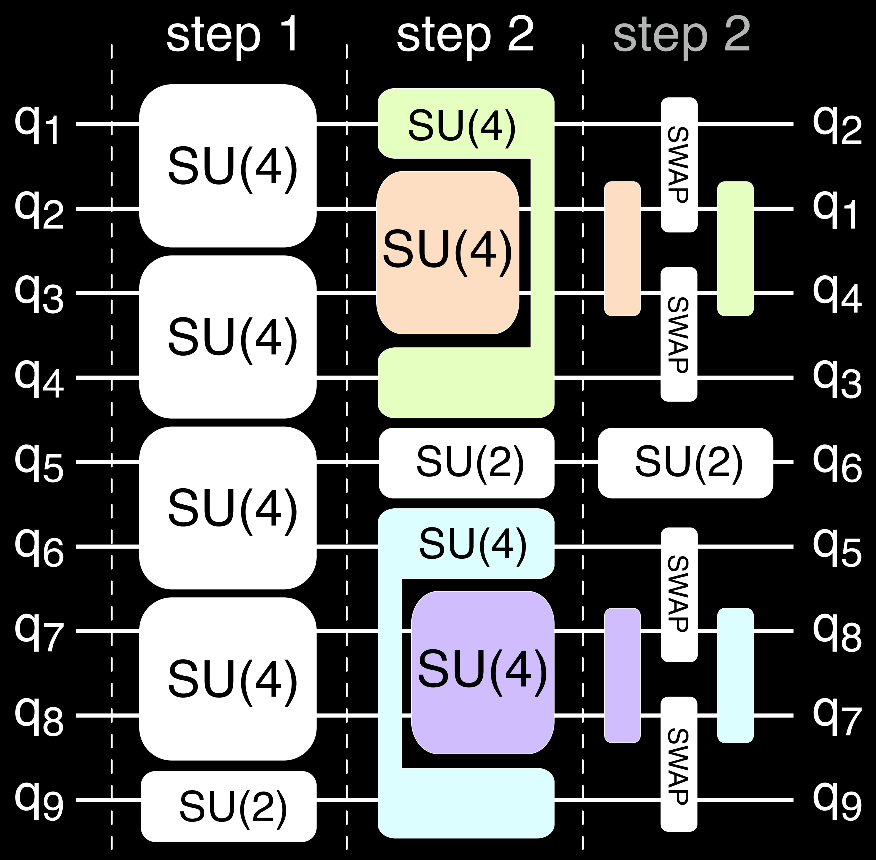
(qubit #) x *(circuit depth)* => $\tilde{V}_Q = \min\left[N, d(N)\right]^2$ => $V_Q = \max_{n<N}\left(\min\left[n, \frac{1}{n\epsilon_{\text{eff}}(n)}\right]^2\right)$

Measured HW parameters :

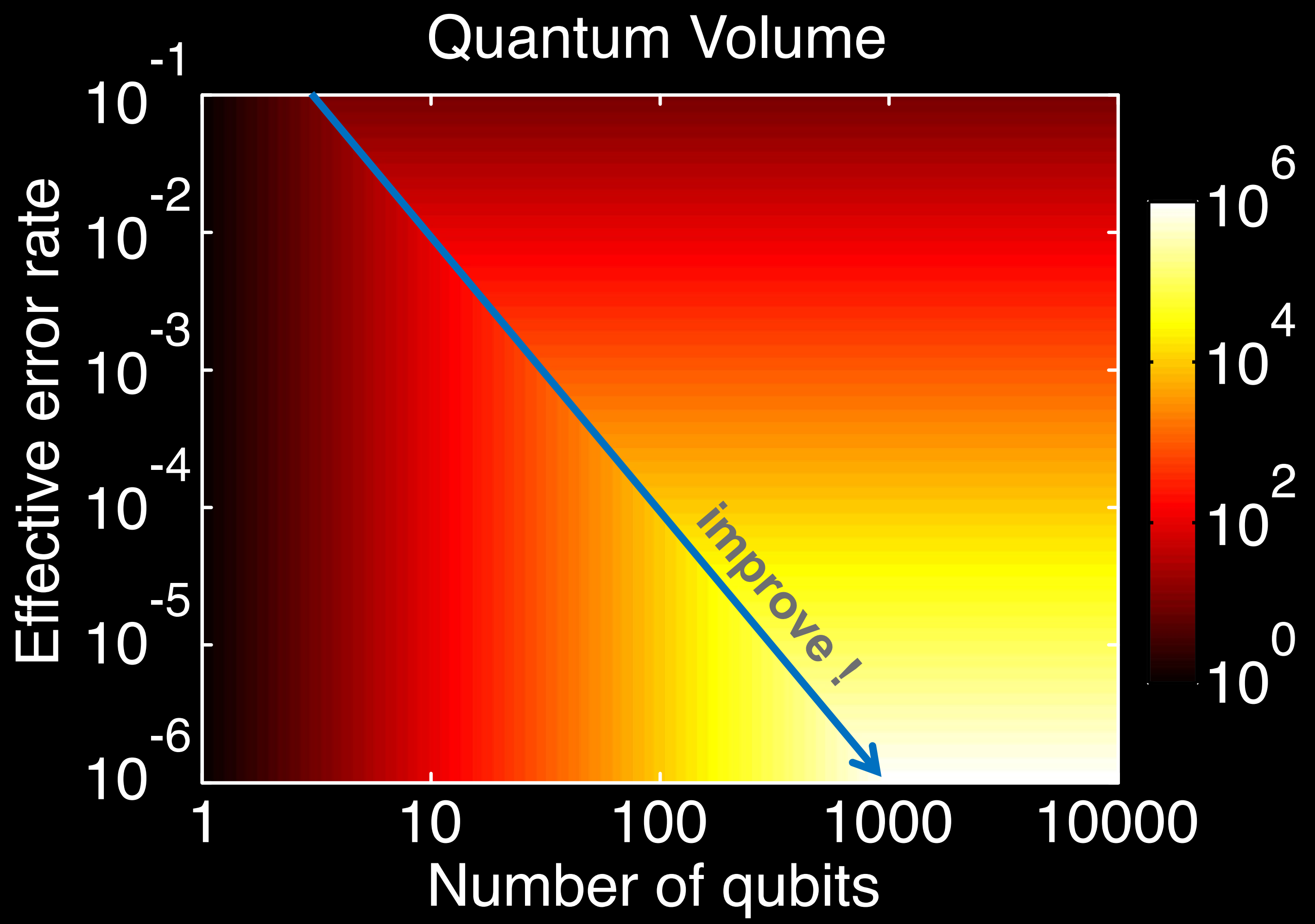
- **Number** of physical qubits N
- **Connectivity** between qubits
- **Number of gates** before errors mask result
- Available **hardware gate set**
- Number of operations that can be run in **parallel**

Measures the useful amount of quantum computing that can be done with a near-term device before errors mask the result.

depth 1 circuit:



Quantum Volume: How powerful is “my” Quantum Computer



Challenges ahead:

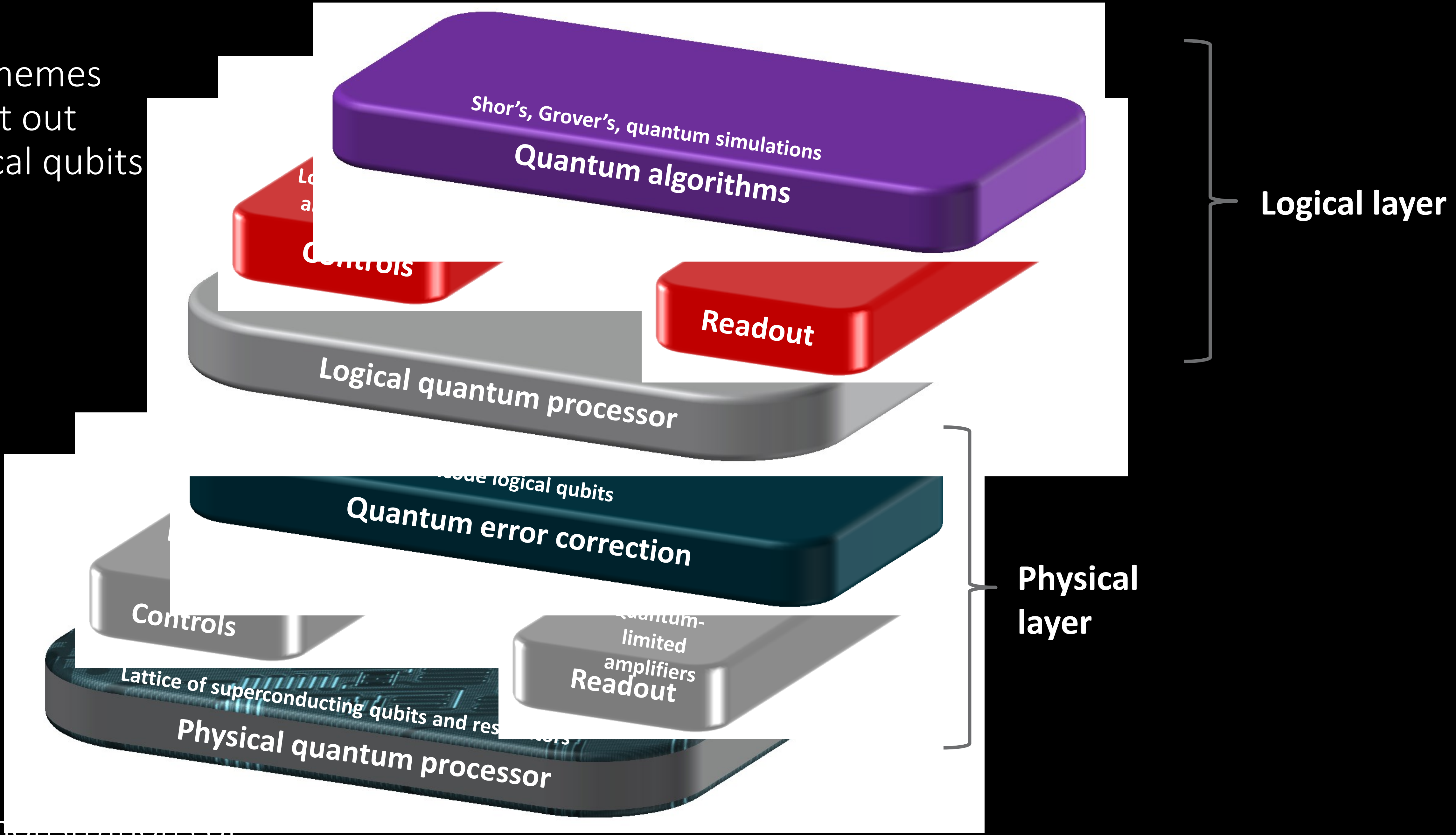
- In order to increase the quantum volume we need to improve ***both*** effective error rate ***and*** qubit number
- It is the scaling of the effective error rate that limits the usefulness of a quantum computer with superconducting qubits
- Need to improve **connectivity, gate speed and coherence**
- Requires better **designs, materials and two qubit interactions**

The Universal Quantum Computing System

Idea:
use error correction schemes
to make a 'logical' qubit out
of several 'good' physical qubits

$$|0\rangle \rightarrow |00\ldots0\rangle$$

$$|1\rangle \rightarrow \underbrace{|11\ldots1\rangle}_{100\text{-}1000\ \times}$$

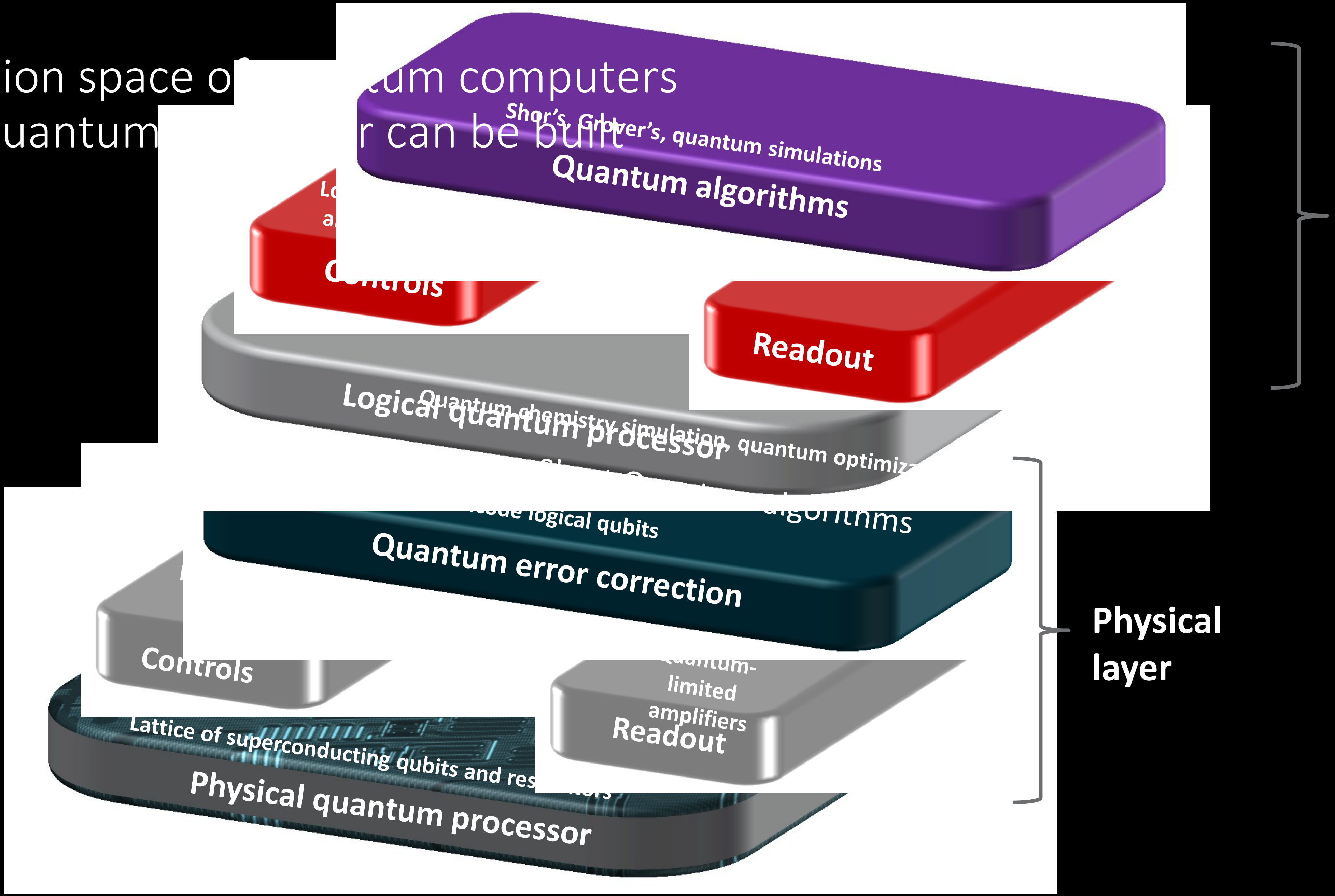


Status:
physical qubits almost
good enough, first demonstrations of
logical qubits expected soon

[Gambetta, Chow, Steffen, npj Quantum Information 3, 2 (2017)]

Approximate Quantum Computing

Explore the application space o tum computers
before a universal quantum r can be built



Logical layer

Physical layer

[Gambetta, Chow, Steffen, npj Quantum Information 3, 2 (2017)]

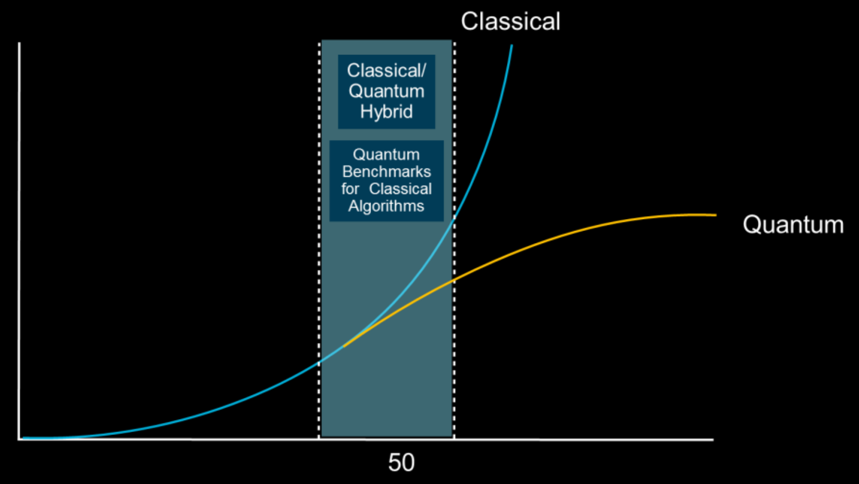
Steps towards Universal Quantum Computing

time / complexity

Demonstration of Quantum Advantage & Learning

Demonstrate an advantage to using quantum computing for real problems of interest. Create software layer, quantum algorithms and education tools.

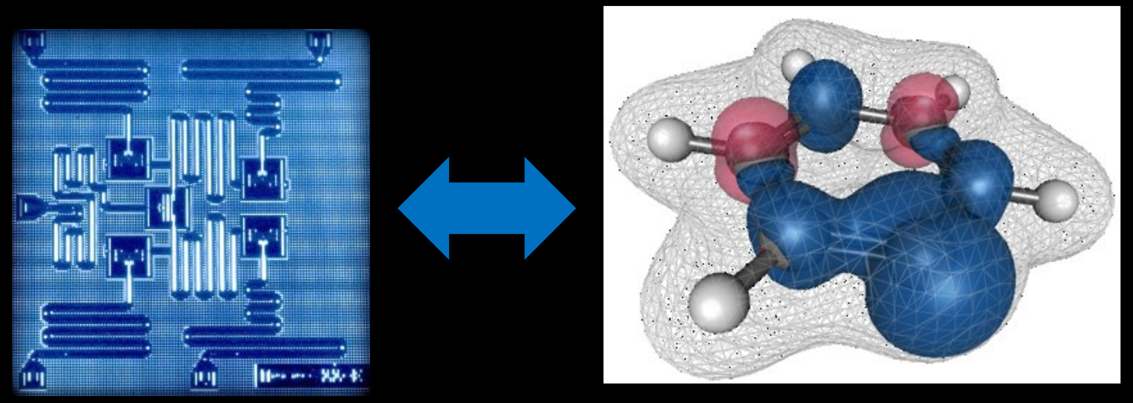
50-100 qubits



Commercialization of Approximate Quantum Computer

Have commercial impact with useful applications on a quantum computer which does not need full fault tolerance, potentially assisting conventional computers (hybrid quantum computer)

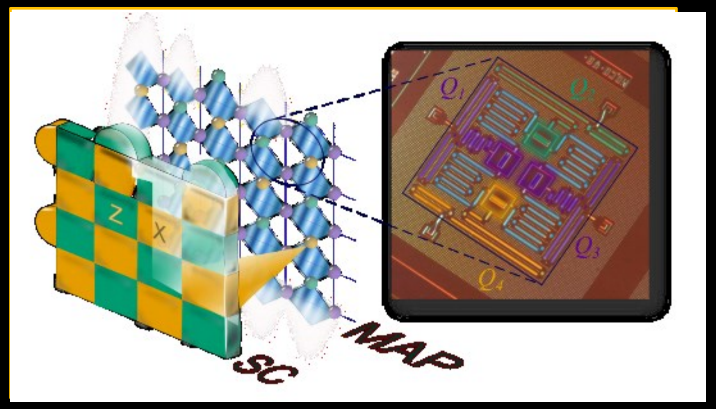
100-1000+ qubits



Universal Fault-Tolerant Quantum Computer

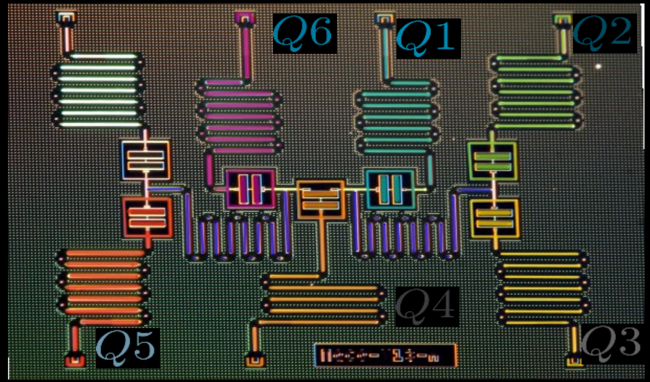
Run useful quantum algorithms with exponential speed up over their classical counterparts. Requires error correction.

1M-10M qubits



Variational Quantum Eigensolver (VQE): A hybrid quantum-classical (HPC) algorithm

Solve problems where the goal is to minimize the energy of a system,
e.g. $E_{min} = \langle\psi(\theta_{min})|H_e|\psi(\theta_{min})\rangle$



Prepare a quantum state $|\psi(\theta)\rangle$
and compute its energy $E(\theta)$

Evaluate $\boldsymbol{E(\theta)}$; use classical optimizer
to choose new value of θ until $\boldsymbol{E_{min}}$ is found

Advantages:

Use short circuits which fit into our coherence time

Improve on best classical estimates by using non-classical trial states

Quantum chemistry

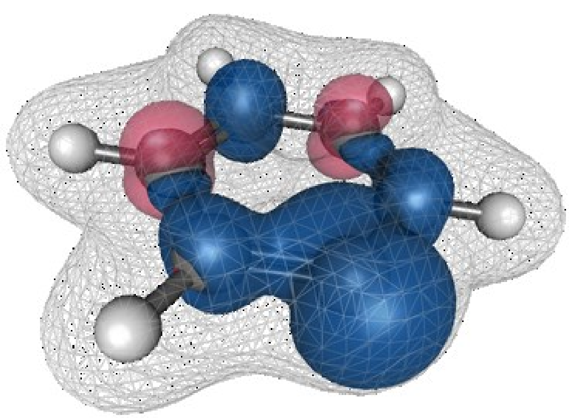
Solving interacting fermionic problems is at the core of most challenges in computational physics and high-performance computing:

$$H_e = -\sum_{i=1}^{N}\frac{1}{2}\nabla_i^2 - \sum_{i=1}^{N}\sum_{A=1}^{M}\frac{Z_A}{r_{iA}} + \sum_{i=1,j,i>1}\frac{1}{r_{ij}}$$

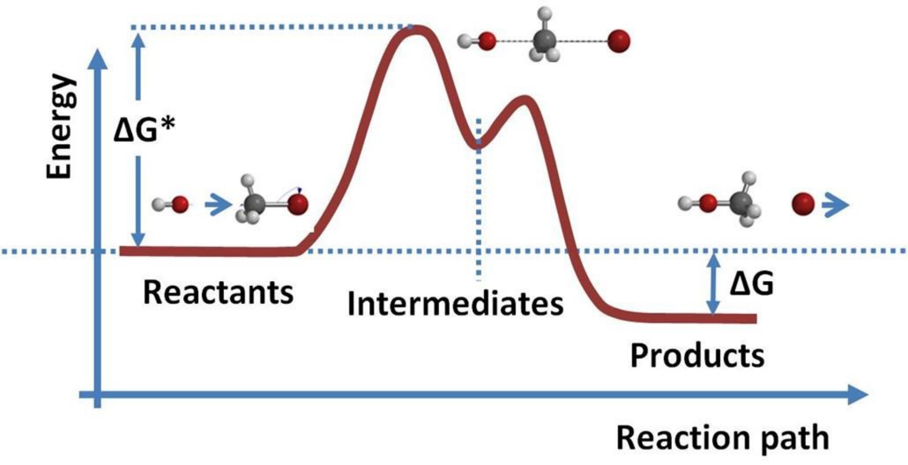
What can quantum computers do?

Map fermions (electrons) to qubits and compute

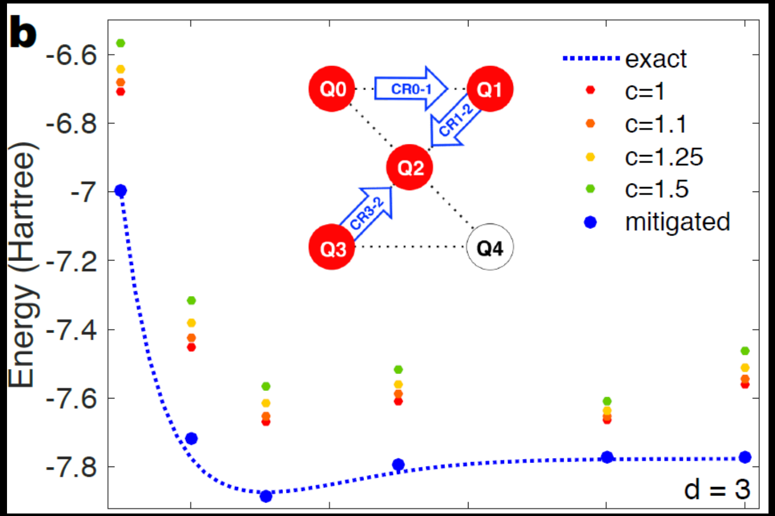
molecular structure



reaction rates



First Demonstrations:

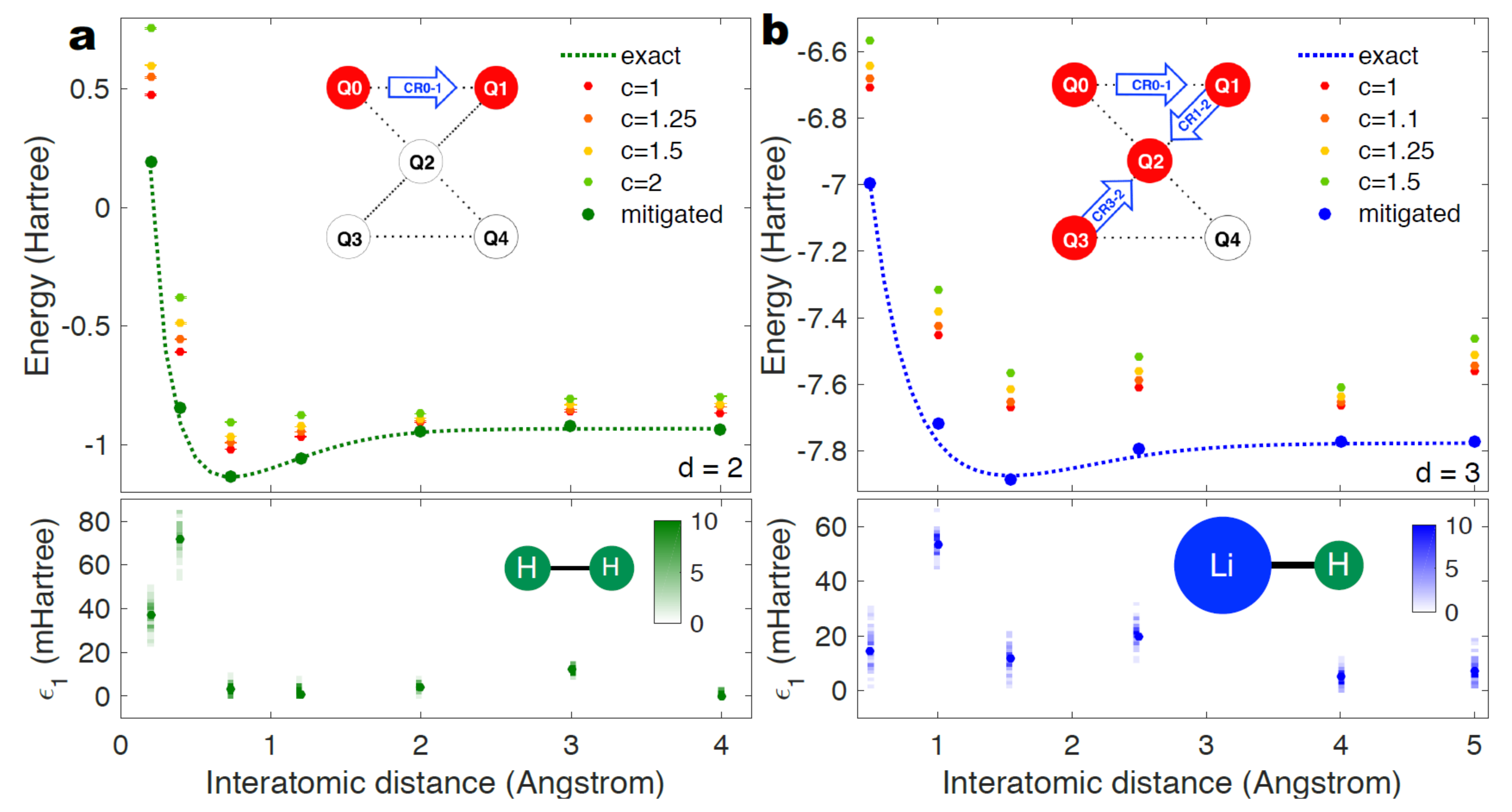


LiH – Molecule (error mitigated)

Sign problem: Monte-Carlo simulations of fermions are NP-hard [Troyer &Wiese, PRL 170201 (2015)]

A. Kandala, et al. Nature 549 (2017); arxiv 1805.04492 (2018)

Groundstate-energy of simple molecules



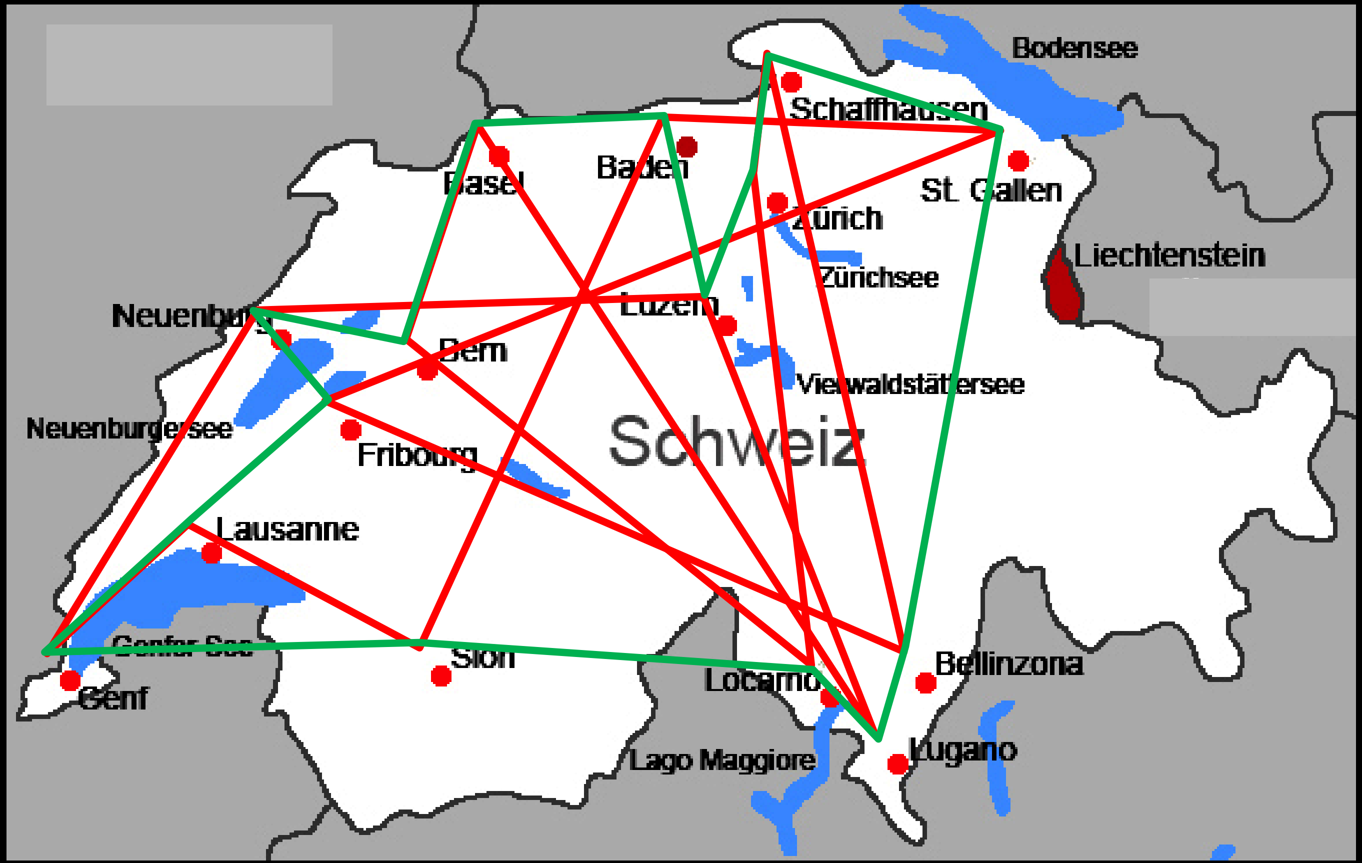
A. Kandala, et al. arXiv:1805.04492 (2018)

Optimization

Traveling Salesman Problem:

- Visit all cities just once
- Choose the shortest path
- Come back to starting point

17 x … x 5 x 4 x 3 x 2 x 1 = 17! =
355’687’428’096’000 possible paths



can be encoded into a quantum physics problem: find the ground-state of a spin (Ising) system, which encodes the optimal path

$$H_{Ising} = \sum_{k=1}^{N} h_k \, \sigma_z^k + \sum_{k<l}^{N} \alpha_{kl} \, \sigma_z^k \otimes \sigma_z^l$$

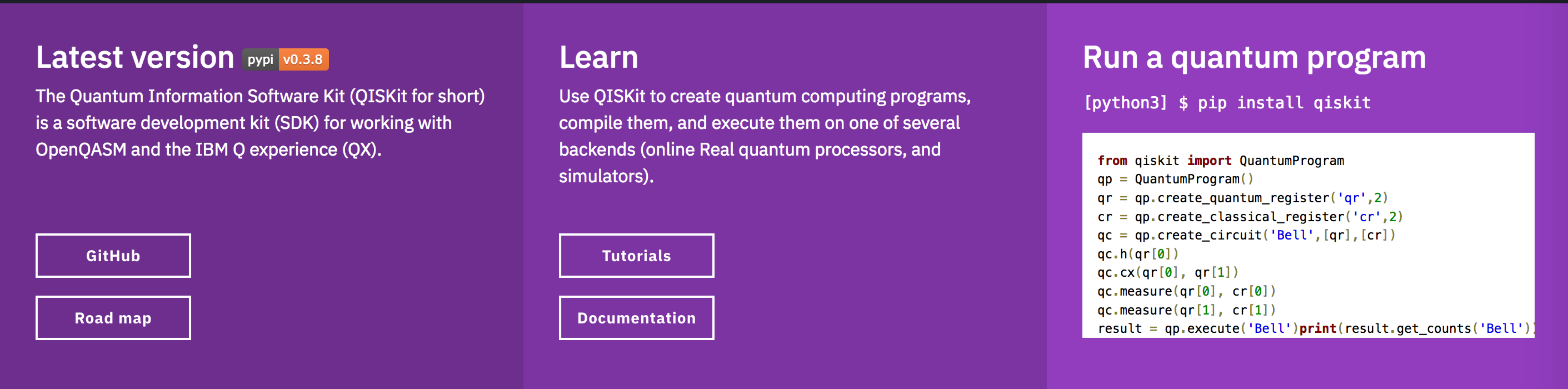
● **18 selected cities in Switzerland**

Optimization Problems & Applications

matching of molecular structures
risk management
traveling salesman problem
forecasting
finite element method
3-SAT
scheduling
portfolio optimization
facility location planning
max cut
manufacturing
image classification
inventory management
capacity planning
production planning
supply chain management
clique
mechanical design
graph coloring
independent set
principal component analysis
customer segmentation
traffic routing
linear programming
data mining
integrated circuit design
energy control

QISKIT Quantum Information Software Kit

Open Source: www.qiskit.org



Latest version

The Quantum Information Software Kit (QISKit for short) is a software development kit (SDK) for working with OpenQASM and the IBM Q experience (QX).

Learn

Use QISKit to create quantum computing programs, compile them, and execute them on one of several backends (online Real quantum processors, and simulators).

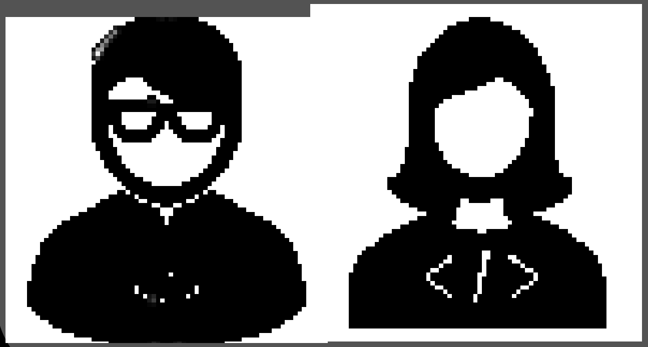
Run a quantum program

```
[python3] $ pip install qiskit
```

```
from qiskit import QuantumProgram
qp = QuantumProgram()
qr = qp.create_quantum_register('qr',2)
cr = qp.create_classical_register('cr',2)
qc = qp.create_circuit('Bell',[qr],[cr])
qc.h(qr[0])
qc.cx(qr[0], qr[1])
qc.measure(qr[0], cr[0])
qc.measure(qr[1], cr[1])
result = qp.execute('Bell')print(result.get_counts('Bell'))
```

From Quantum Experience to Quantum Programs

Quantum Developers



Build

Python Interface

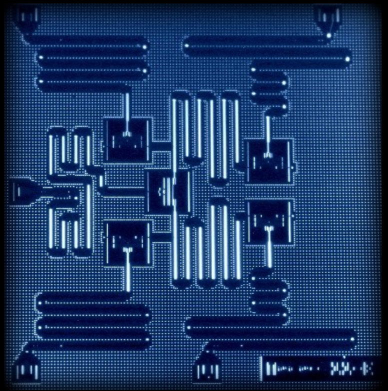
Compile

Translate & Optimize

API

Experience

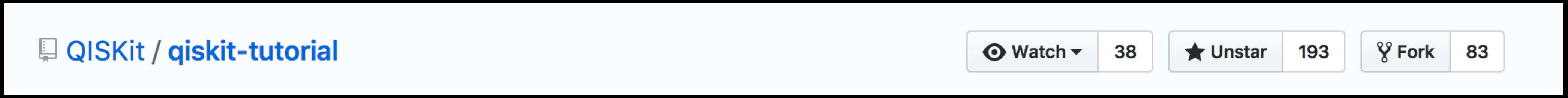
Real Devices

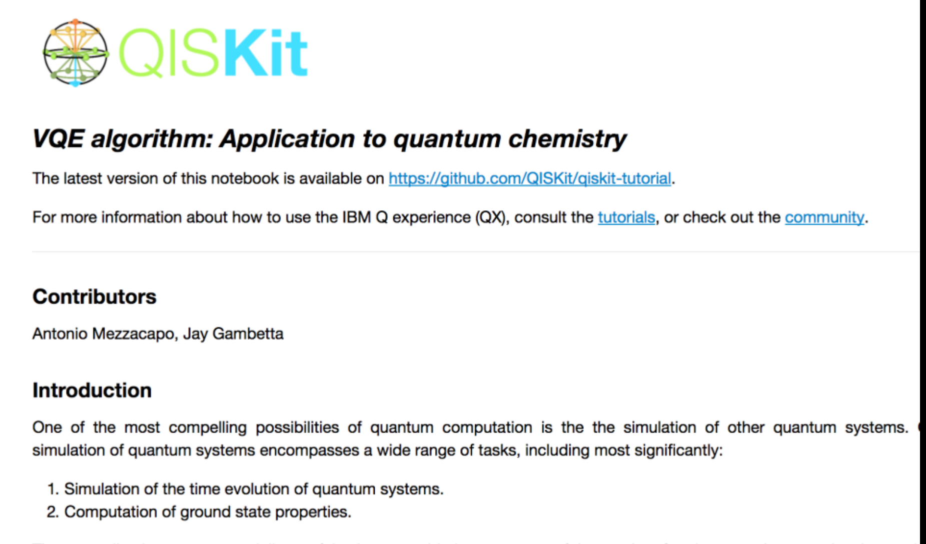


Simulators

Laboratory

QISKit demos for quantum chemistry and optimization


QISKit / **qiskit-tutorial**


VQE algorithm: Application to quantum chemistry

The latest version of this notebook is available on https://github.com/QISKit/qiskit-tutorial.

For more information about how to use the IBM Q experience (QX), consult the tutorials, or check out the community.

Contributors

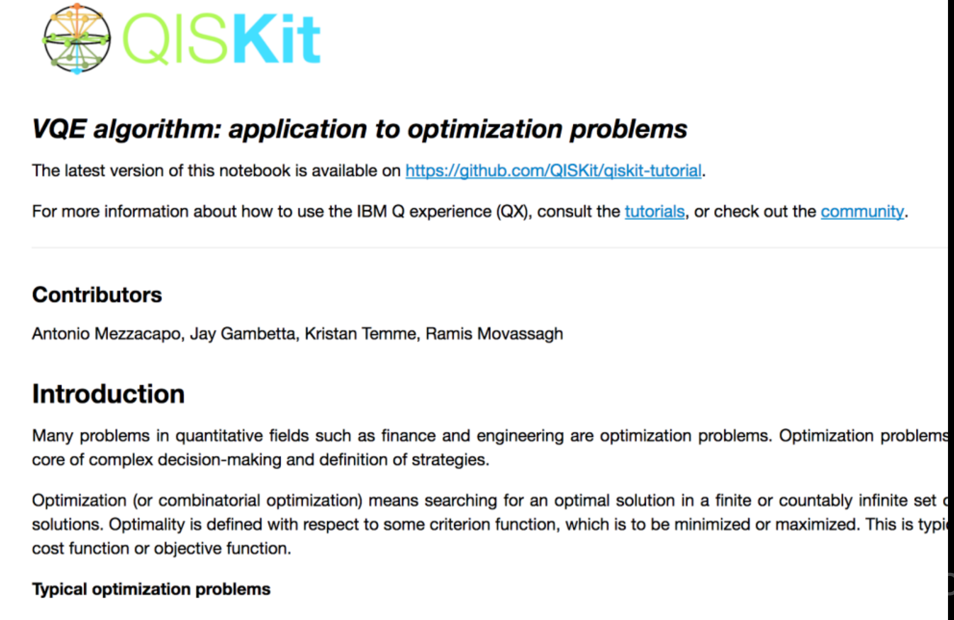
Antonio Mezzacapo, Jay Gambetta

Introduction

One of the most compelling possibilities of quantum computation is the the simulation of other quantum systems. simulation of quantum systems encompasses a wide range of tasks, including most significantly:

1. Simulation of the time evolution of quantum systems.
2. Computation of ground state properties.

These applications are especially useful when considering systems of interacting fermions, such as molecules and correlated materials. The computation of ground state properties of fermionic systems is the starting point for mapping out t


VQE algorithm: application to optimization problems

The latest version of this notebook is available on https://github.com/QISKit/qiskit-tutorial.

For more information about how to use the IBM Q experience (QX), consult the tutorials, or check out the community.

Contributors

Antonio Mezzacapo, Jay Gambetta, Kristan Temme, Ramis Movassagh

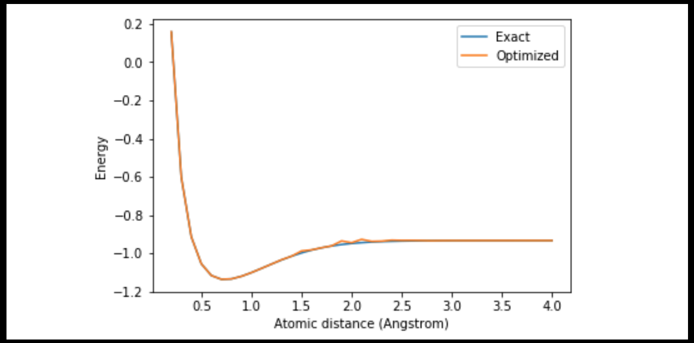
Introduction

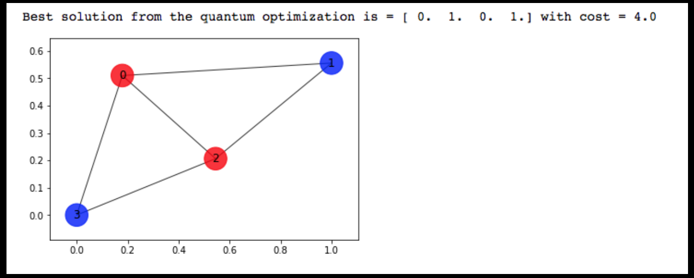
Many problems in quantitative fields such as finance and engineering are optimization problems. Optimization problem core of complex decision-making and definition of strategies.

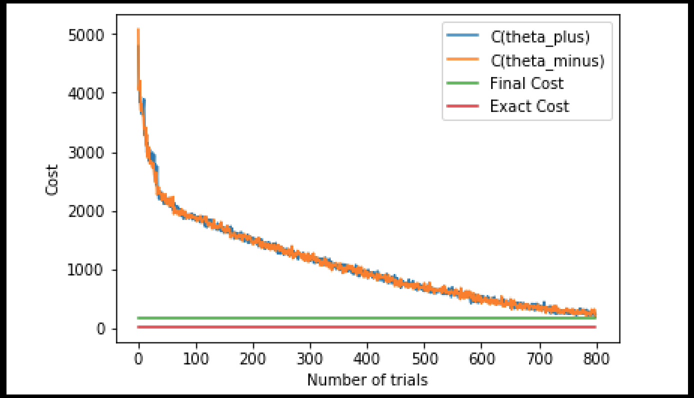
Optimization (or combinatorial optimization) means searching for an optimal solution in a finite or countably infinite set solutions. Optimality is defined with respect to some criterion function, which is to be minimized or maximized. This is typi cost function or objective function.

Typical optimization problems

- Step by step Jupiter notebooks on how to run quantum chemistry and combinatorial problems using quantum computers
- Variational quantum eigensolver approach with short-depth trial wavefunctions
- Chemistry tutorial includes H2 and LiH molecules to be run on 2 and 4 qubits. Explicit run of one interatomic distance and full potential energy surface
- Toy optimization problems include Max-Cut and travelling salesman instances



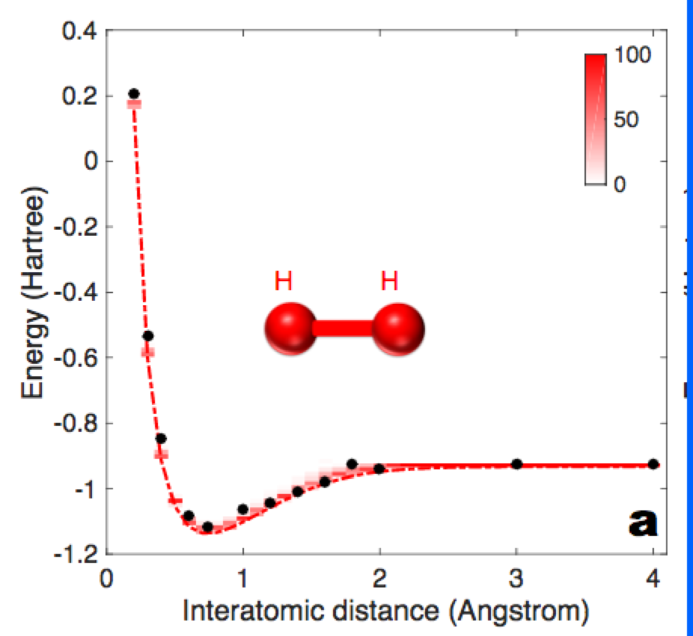

```
Best solution from the quantum optimization is = [ 0.  1.  0.  1.] with cost = 4.0
```



QISkit ACQUA

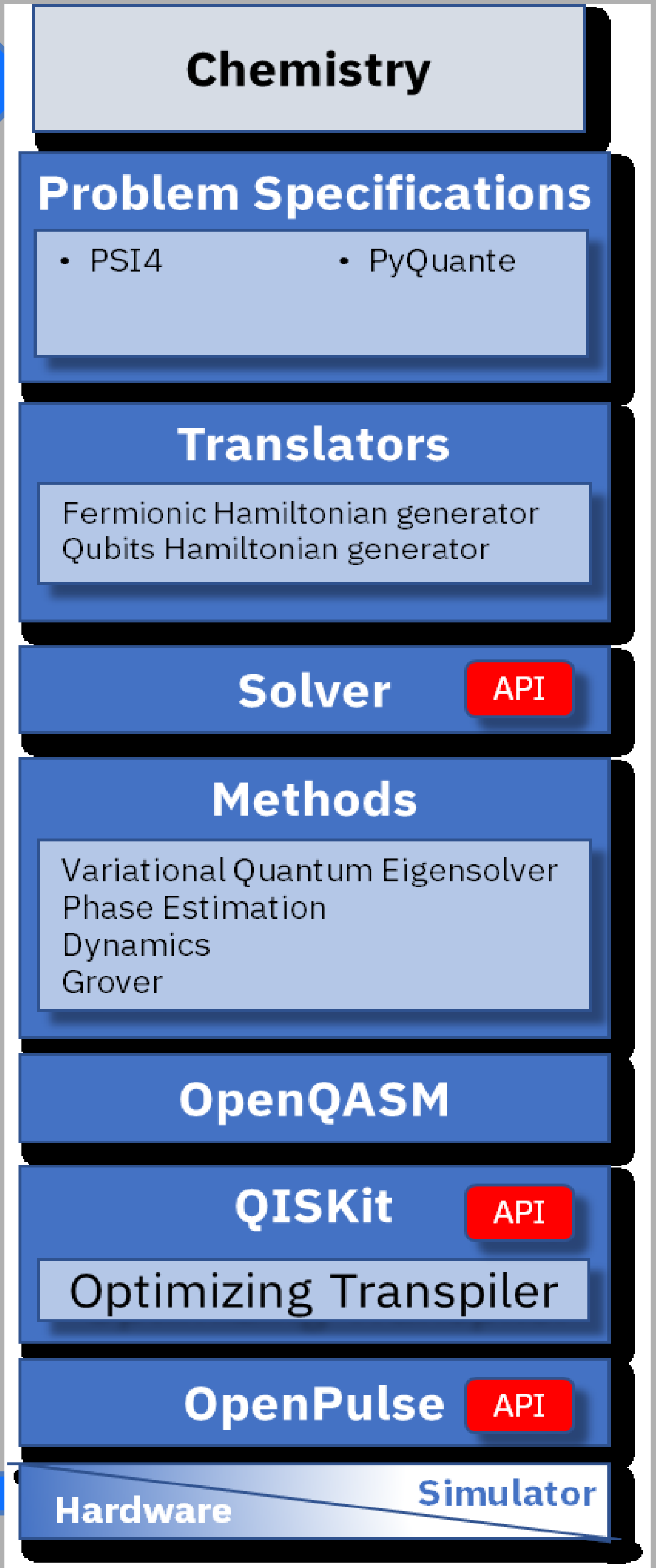
End Users and Domain Experts:

- Seamlessly integrate Quantum capabilities in the existing workflow
- Enjoy Quantum performance and accuracy gains without having to know Quantum
- Easily get high quality results



```
[-1.70505133  5.08016856
  6.13194571  4.12524575
 -0.26812628  3.90229532
  5.60418357 -3.50842279
 -3.92937001  6.17154054
  5.55945014  5.89555566
 -4.83748112 -5.56210097
  5.58561689 -5.99641163
  1.65834199 -1.49309666
 -2.47010646  0.4242964
  0.39126571  0.46051075
  4.28488151 -5.30498883]
```

```
-1.875 Hartree
```



Chemistry

Problem Specifications

- PSI4
- PyQuante

Translators

Fermionic Hamiltonian generator
Qubits Hamiltonian generator

Solver API

Methods

Variational Quantum Eigensolver
Phase Estimation
Dynamics
Grover

OpenQASM

QISKit API

Optimizing Transpiler

OpenPulse API

Hardware **Simulator**

Grand Challenge: Quantum Computing

Goal:

Build computers based on quantum physics to solve problems that are otherwise intractable

Roadmap:

Small-scale (Demonstration of Quantum advantage)

- Research level demonstrations
- Verify chemistry and error correction principles
- Infrastructure & community building
- Demonstrate 'Quantum advantage'

Medium-scale (Commercializing approximate QC)

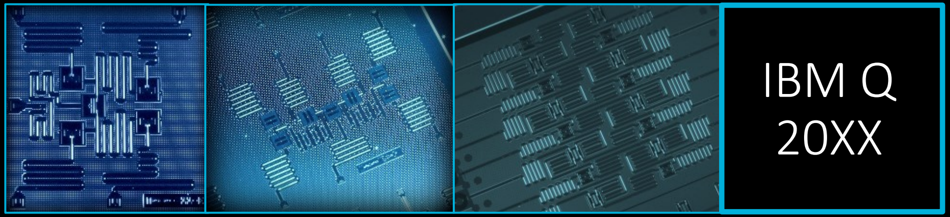
- Develop "Hardware-efficient" apps
  - Chemical configurations
  - Optimization
  - Hybrid quantum-classical computers
- No full error correction available

Large-scale (Fault-tolerant Universal QC)

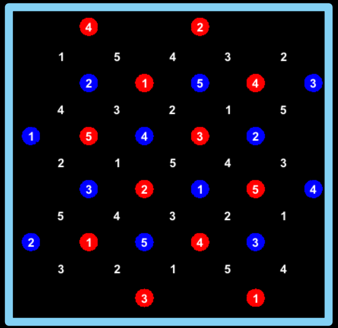
Known and proven speed-up:

- Factoring
- quantum molecular simulations
- Speed-up machine learning

Enable secure cloud computing



5-8 qubits 16 qubits 50+100 qubits



100-1000+ qubits

10^6 -10^7 qubits