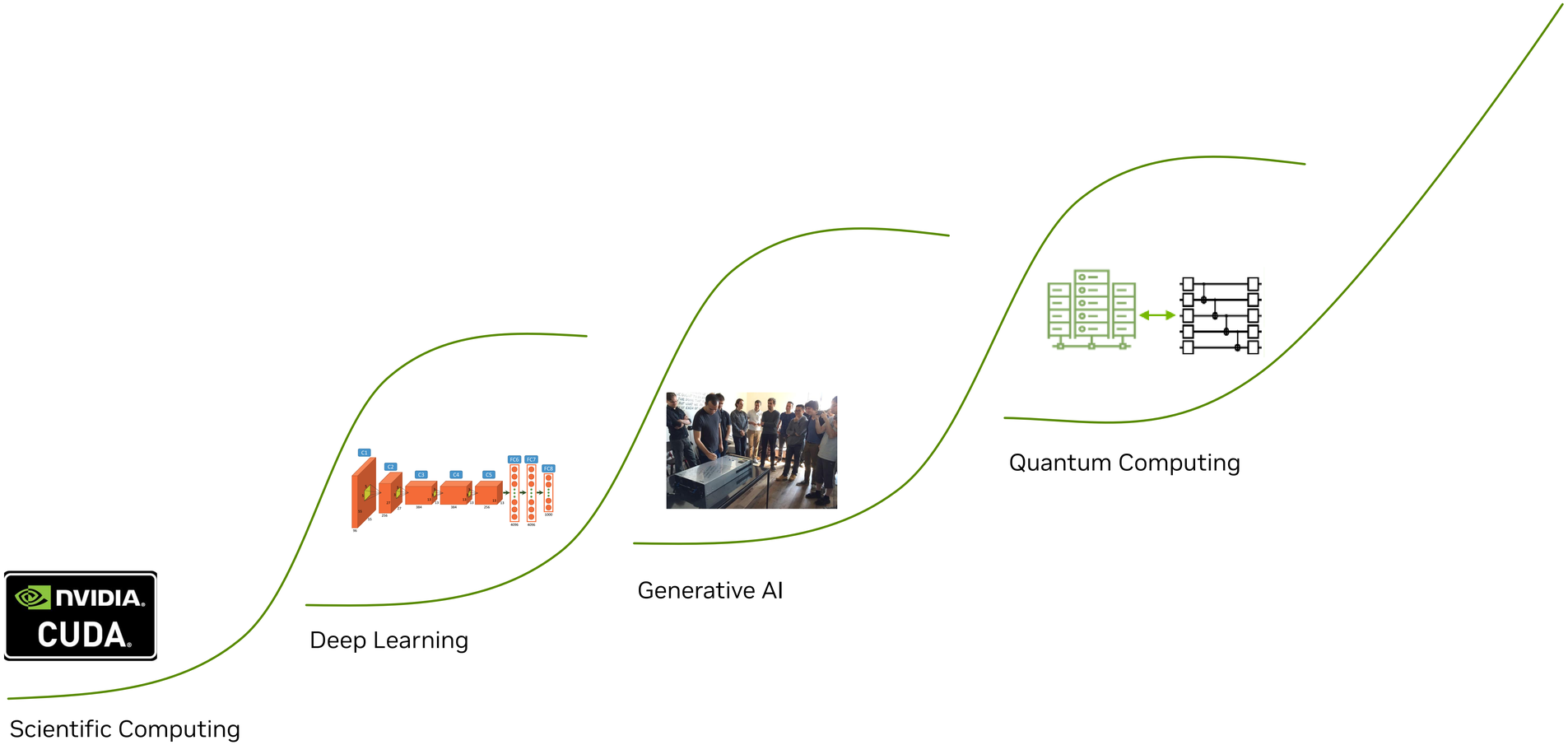
Programming Quantum-Accelerated Supercomputers with CUDA Quantum

Esperanza Cuenca Gómez

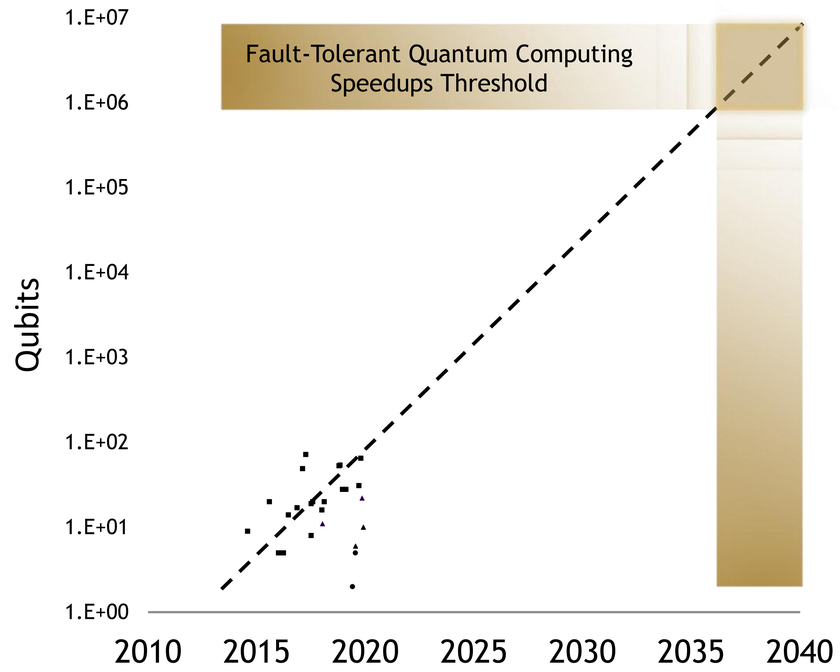
Developer Relations Manager, Quantum Computing

NVIDIA History

Decades of leadership in accelerated computing



Challenges for Useful Quantum Computing



SCALE

100,000+

Total Qubits Required

ALGORITHMS

Unknown

Algorithms for Advantage

FIDELITY

1,000+

Perfect Error Corrected
Logical Qubits Required

INTEGRATION

Open

Challenge to integrate Quantum
in with Classical Supercomputing

NVIDIA QUANTUM

Empowering the Quantum Computing Community

QUANTUM HARDWARE BUILDERS

ANYON · atom computing · Google Quantum AI · Infleqtion · IONQ

IQM · OQC · ORCA COMPUTING · PASQAL · PsiQuantum · QUANTINUUM

QUANTUM BRILLIANCE · QuEra COMPUTING INC. · rigetti · seeqc · XANADU

QUANTUM SOFTWARE AND SYSTEMS

agnostiq · CLASSIQ · HQS QUANTUM SIMULATIONS · KIPU QUANTUM · menten.AI

MULTIVERSE COMPUTING · OTI · QCWARE · KILIMANJARO QUANTUM TECH · QM ware

QUANTUM MACHINES · Qubit pharmaceuticals · STRANGEWORKS · ZAPATA

QUANTUM SIMULATION FRAMEWORKS

aws · blueqat · Cirq

Orquestra® · PENNYLANE · QIBO

Qiskit · TensorFlow Quantum · XACC QUANTUM FRAMEWORK

ENTERPRISE PARTNERS

DELL Technologies · Deloitte. · Hewlett Packard Enterprise · softserve

aws · Google Cloud · Microsoft Azure · ORACLE CLOUD Infrastructure

Johnson & Johnson · VW · BMW · GE · SIEMENS Healthineers

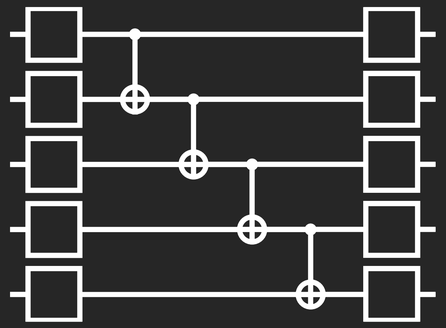
RESEARCH CENTERS

ABCI AI Bridging Cloud Infrastructure · Argonne NATIONAL LABORATORY · cea · CINECA · GENCI Le calcul intensif au service de la connaissance · JÜLICH Forschungszentrum · NCSA · NERSC · OAK RIDGE National Laboratory · RIKEN

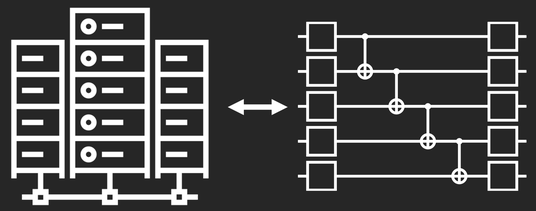
NVIDIA Quantum

Powering Quantum Simulation and Quantum-Integrated Accelerated Computing

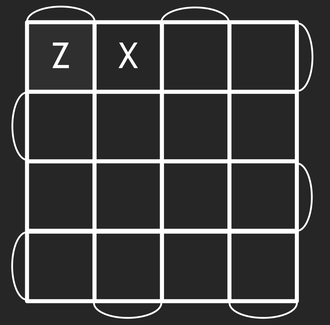
Quantum Algorithms Research



Quantum-Integrated Applications



Error Correction, Calibration, Control



cuQuantum
Accelerated Quantum Simulation

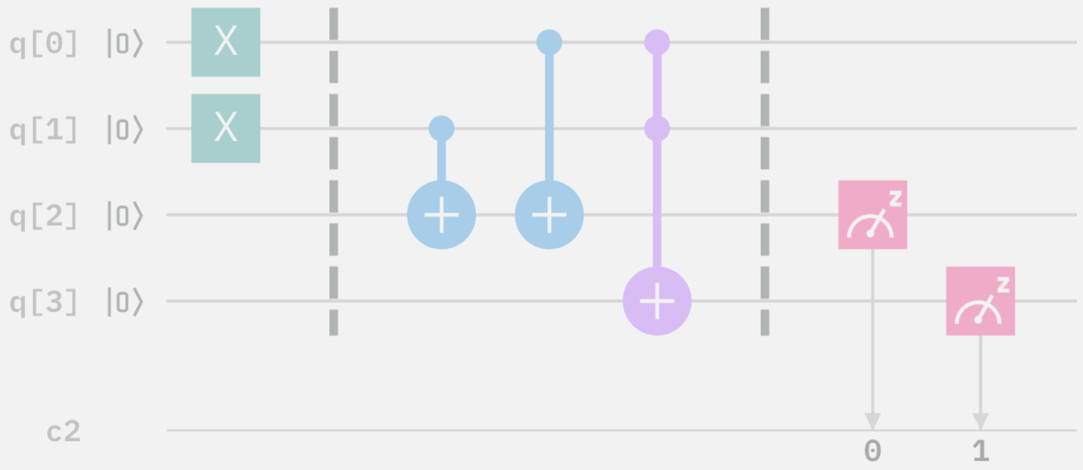
CUDA Quantum
Quantum-Classical Developer Platform

Quantum Integrated GPU Supercomputing
DGX | HGX | DGX Quantum

GPU Supercomputing and Quantum

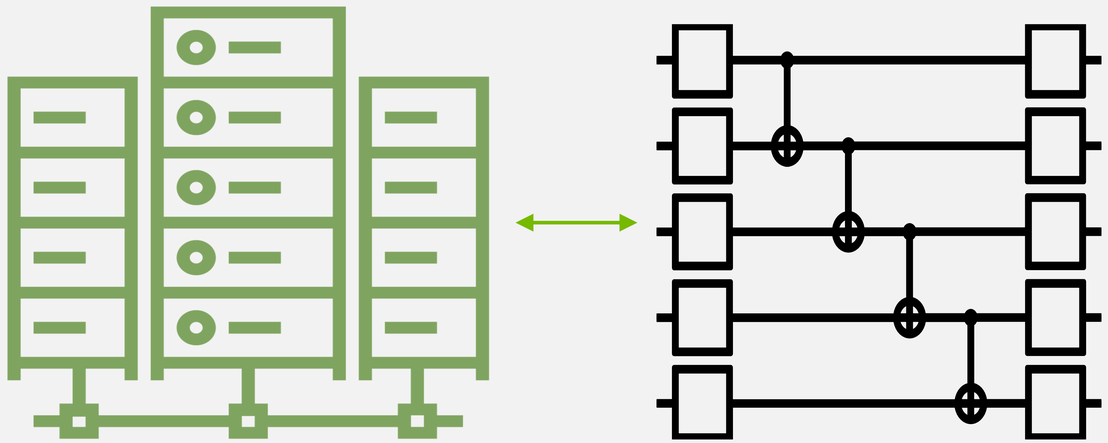
Researching the Quantum Computers of Tomorrow with the Supercomputers of Today

QUANTUM SIMULATION



- Develop algorithms at scale of valuable quantum computing
- Discover use cases with quantum advantage
- Design and validate future hardware

HYBRID QUANTUM-CLASSICAL COMPUTING

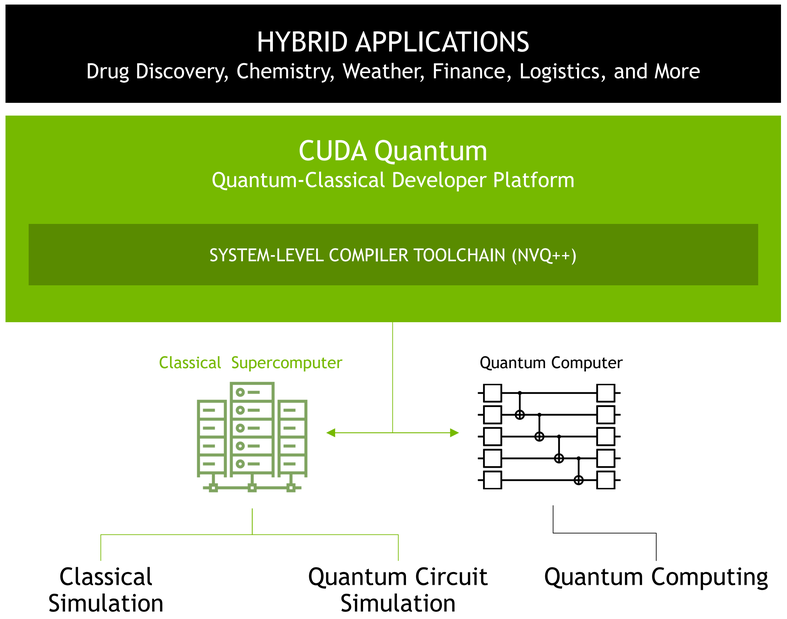


- Integrate quantum into leading accelerated applications
- Unparalleled performance and scientific productivity using the best resource for the task
- GPUs critical for QEC, calibration, hybrid algorithms

CUDA Quantum

A Platform For Quantum-Classical Computing

CUDA QUANTUM PLATFORM



CUDA QUANTUM FEATURES

Supports any kind of QPU, emulated or physical

Compiler for hybrid systems

Open and interoperable with today's applications

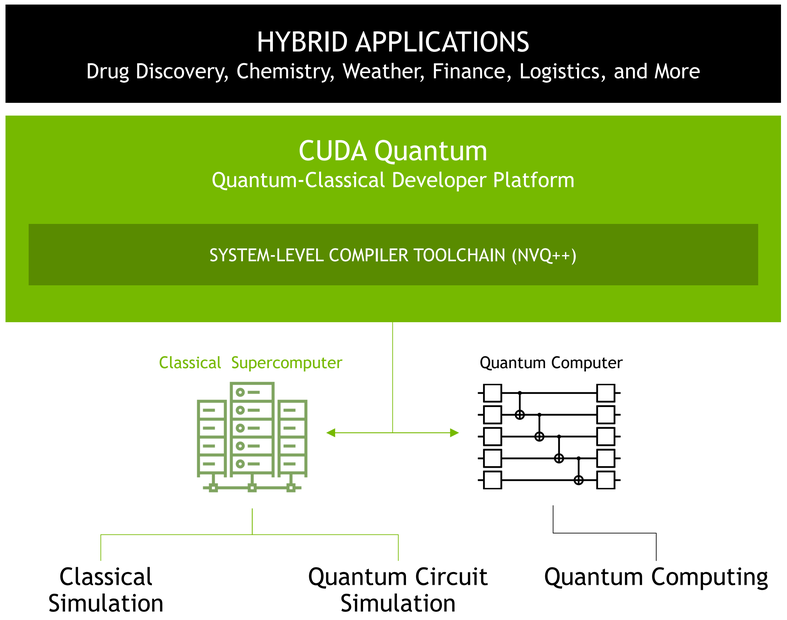
Single source C++ and Python programming model

github.com/nvidia/cuda-quantum | https://catalog.ngc.nvidia.com/orgs/nvidia/containers/cuda-quantum

CUDA Quantum

Adopted by Community's Leaders to Enable Quantum-Accelerated Applications

CUDA QUANTUM PLATFORM



CUDA QUANTUM PARTNERS

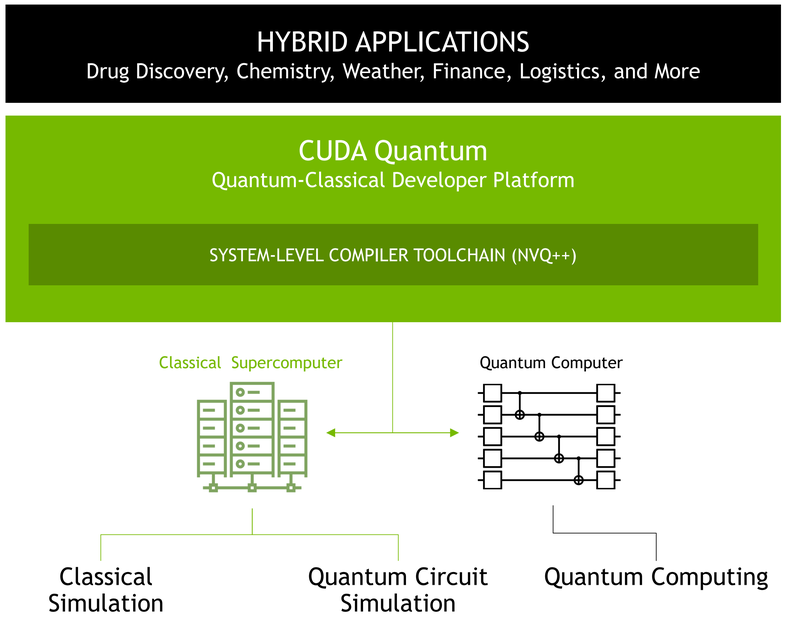
ANYON · atom computing · IONQ · IQM · OQC · ORCA COMPUTING · PASQAL · QUANTUM BRILLIANCE · QUANTINUUM · QuEra COMPUTING INC. · rigetti · XANADU · agnostiq · CLASSIQ · Microsoft · QCWARE · QM ware · ZAPATA

github.com/nvidia/cuda-quantum | https://catalog.ngc.nvidia.com/orgs/nvidia/containers/cuda-quantum

CUDA Quantum

Natively Hybrid And Interoperable With GPU Supercomputing

CUDA QUANTUM PLATFORM



Interoperable with GPU Supercomputing

```
auto cnts = cudaq::sample(q, …);
```

Standard Parallelism

```
std::sort(std::execution::par, …);
```

CUDA

```
kernel<<<…>>>(…);
cudaDeviceSynchronize();
```

OpenMP

```
#pragma omp target teams loop
for (…) …
```

OpenACC

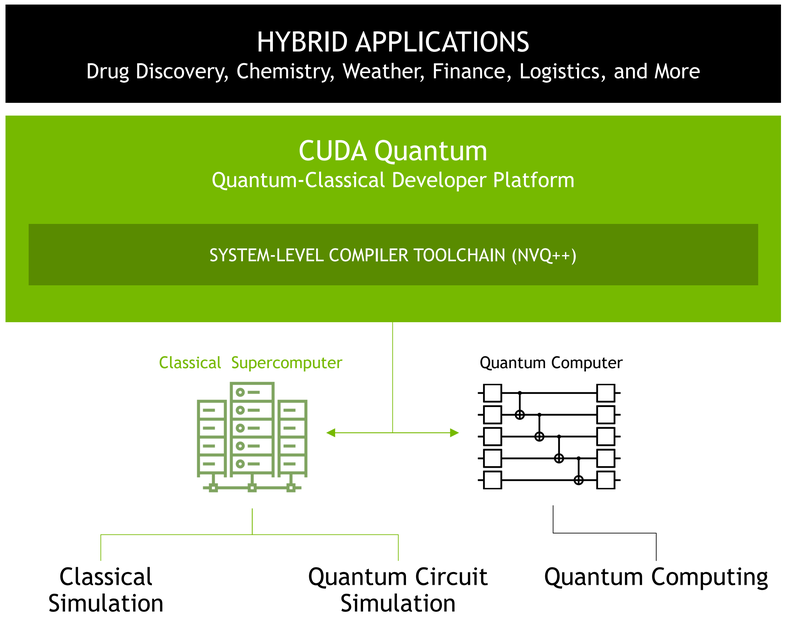
```
#pragma acc parallel loop
for (…) …
```

github.com/nvidia/cuda-quantum | https://catalog.ngc.nvidia.com/orgs/nvidia/containers/cuda-quantum

CUDA Quantum

Natively Hybrid And Interoperable With GPU Supercomputing

CUDA QUANTUM PLATFORM



Interoperable with GPU Supercomputing

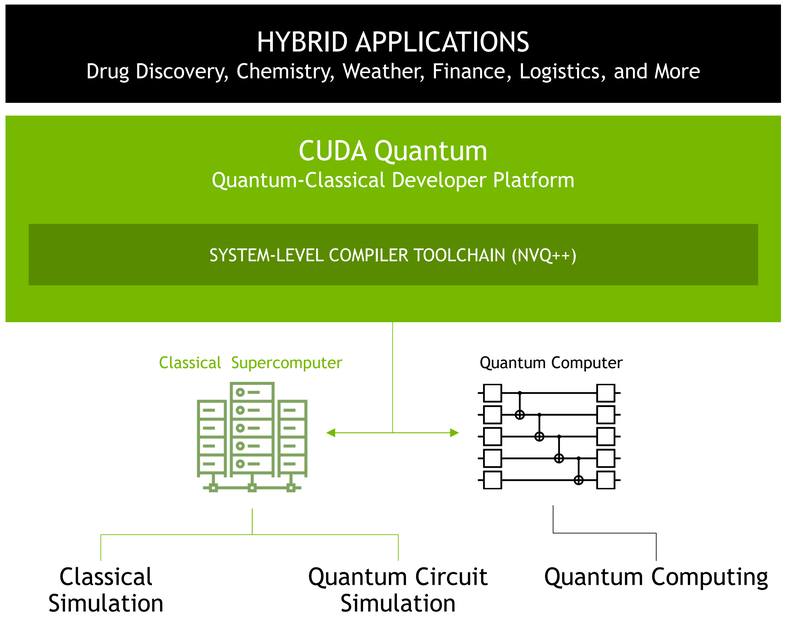
```
// Compute expectation values with QPU.
cudaq::spin_op h = ...;
std::vector<double> sig_exps;
for (auto& pauli_op : generate_pauli_permutations(h.n_qubits()))
  sig_exps.push_back(cudaq::observe(qite, pauli_op, ...));



...
// Compute LU Factorization of S_mat on the GPU.
auto dim = std::pow(2, h.n_qubits());
cusolverDnXgetrf(handle, params, dim, dim, CUDA_C_64F, S_mat,
                 lda, NULL, CUDA_C_64F,
buffer_on_device,
                 bytes_on_device, buffer_on_host,
                 bytes_on_host, info);
```

github.com/nvidia/cuda-quantum | https://catalog.ngc.nvidia.com/orgs/nvidia/containers/cuda-quantum

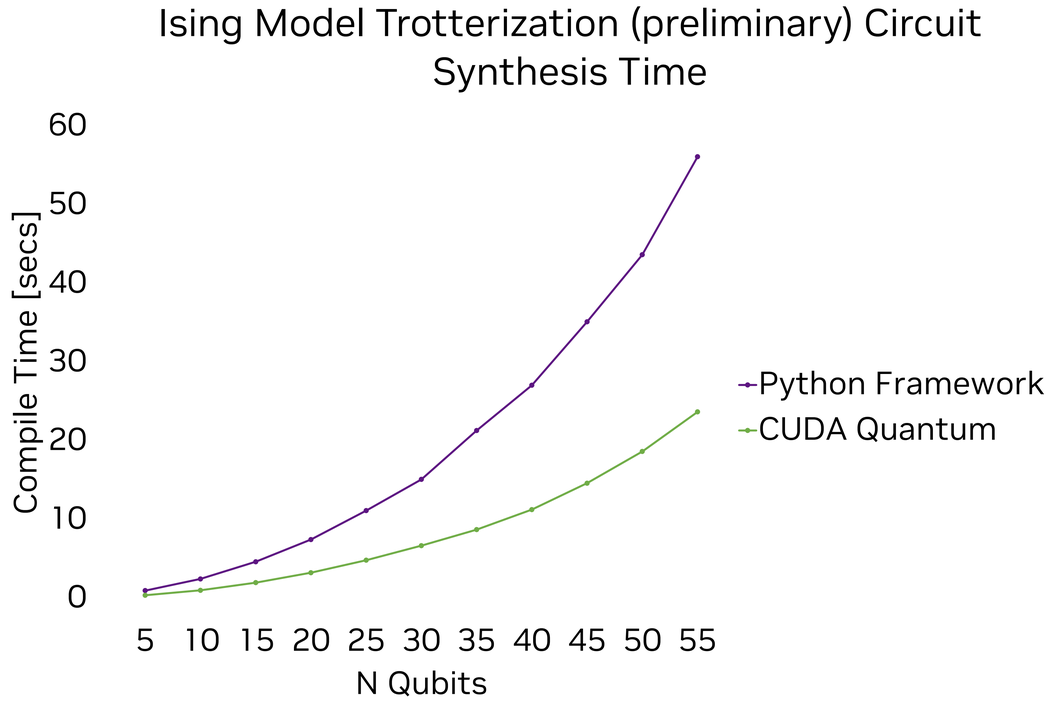
CUDA Quantum

Delivering Unmatched Performance, Scalability, And Usability

CUDA QUANTUM PLATFORM



ENGINEERED FOR PERFORMANCE AND SCALE

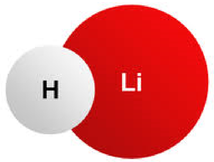


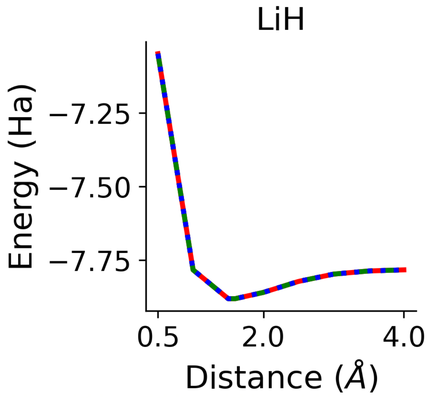
github.com/nvidia/cuda-quantum | https://catalog.ngc.nvidia.com/orgs/nvidia/containers/cuda-quantum

VQE: NVQ++/cuStateVec on A100 GPU vs Leading Pythonic Framework and SImulator with Thrust on A100 GPU

Variational quantum eigensolver (VQE)

$$|\psi(\theta)\rangle$$



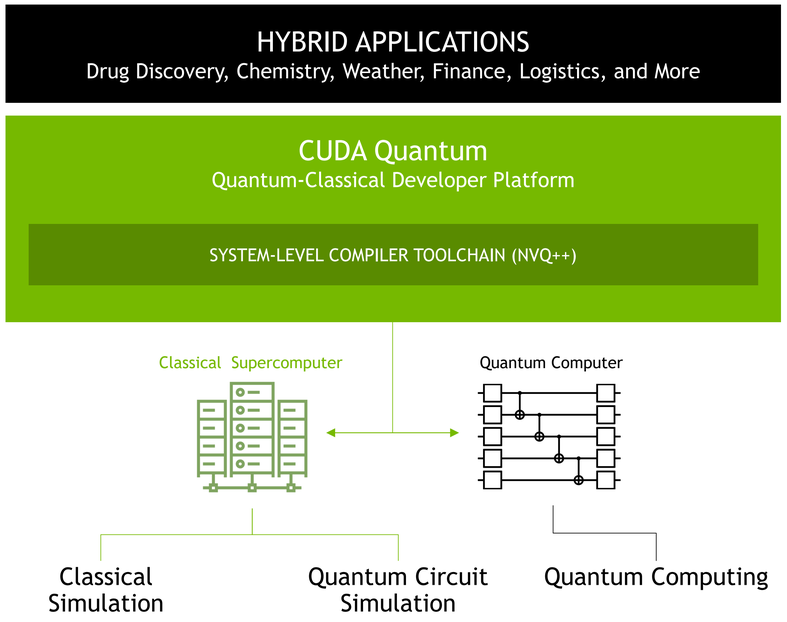


$$\langle\psi(\theta)|H|\psi(\theta)\rangle \approx E_{gs}$$

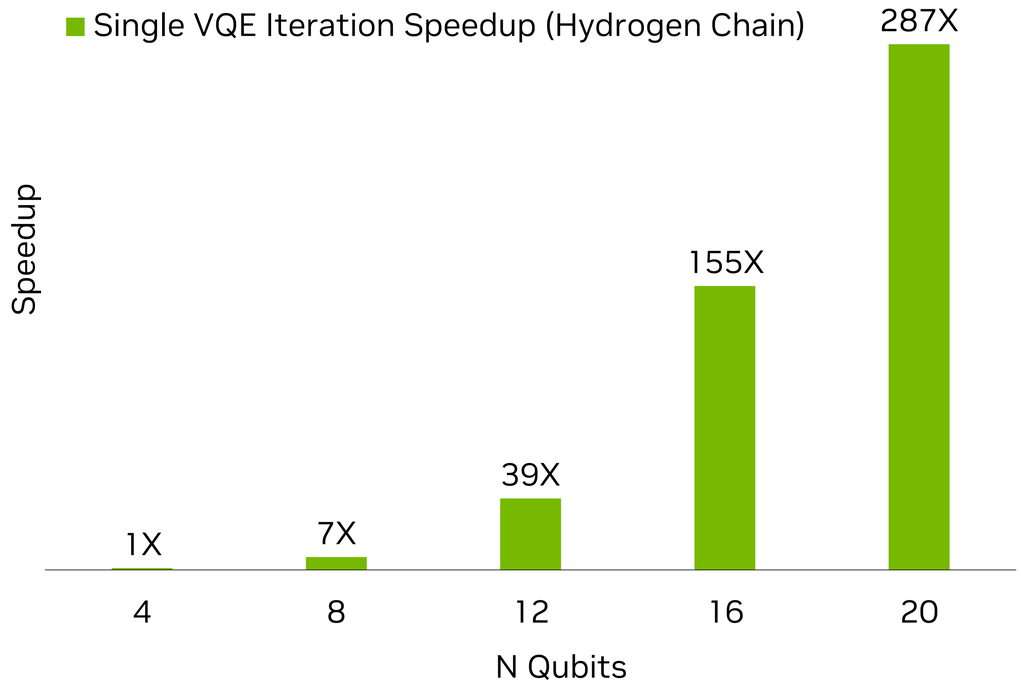
CUDA Quantum: Now Available on GitHub and NGC

Delivering Unmatched Performance, Scalability, And Usability

CUDA QUANTUM PLATFORM



ENGINEERED FOR PERFORMANCE AND SCALE



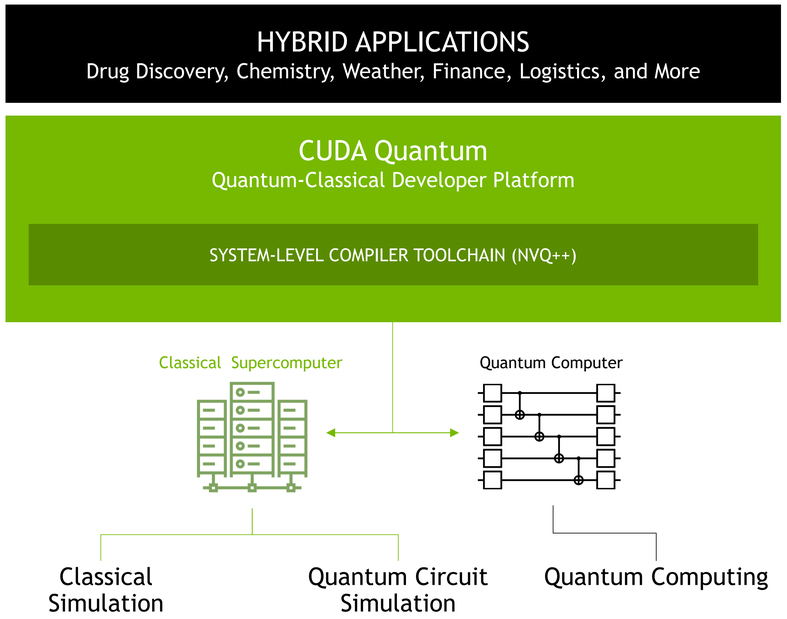
github.com/nvidia/cuda-quantum | https://catalog.ngc.nvidia.com/orgs/nvidia/containers/cuda-quantum

VQE: NVQ++/cuStateVec on A100 GPU vs Leading Pythonic Framework and SImulator with Thrust on A100 GPU

CUDA Quantum: Now Available on GitHub and NGC

Natively Hybrid And Interoperable With GPU Supercomputing

CUDA QUANTUM PLATFORM



Python and C++

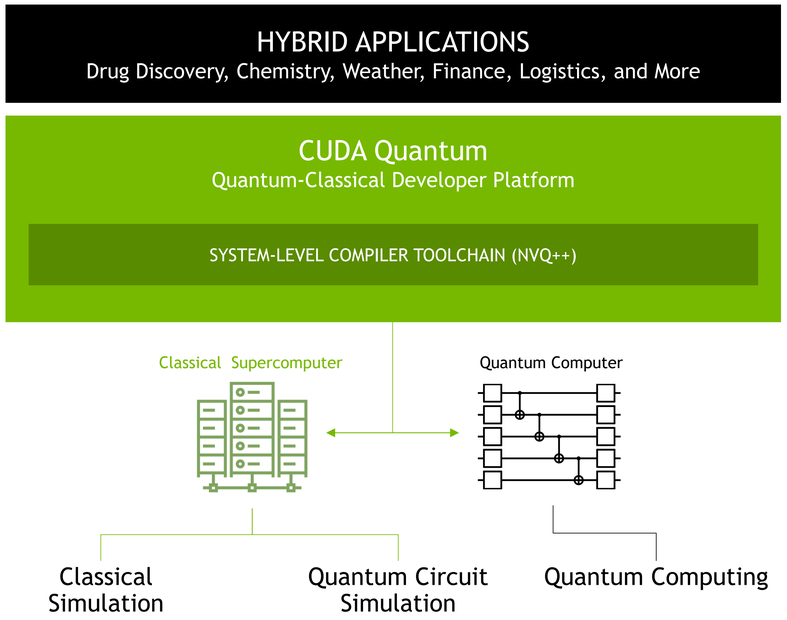
```python
import cudaq

# Set the backend
cudaq.set_qpu('quantinuum')

# Create the kernel function signature
# here void(float)
ansatz, theta = cudaq.make_kernel(float)
q = ansatz.qalloc(2)
ansatz.x(q[0])
ansatz.ry(theta, q[1]);
ansatz.cx(q[0], q[1]);

h = cudaq.SpinOperator(...)

result = cudaq.observe(ansatz, h, .59)
print('<H> = ', result.expectation_z())
```

github.com/nvidia/cuda-quantum | https://catalog.ngc.nvidia.com/orgs/nvidia/containers/cuda-quantum

CUDA Quantum: Now Available on GitHub and NGC

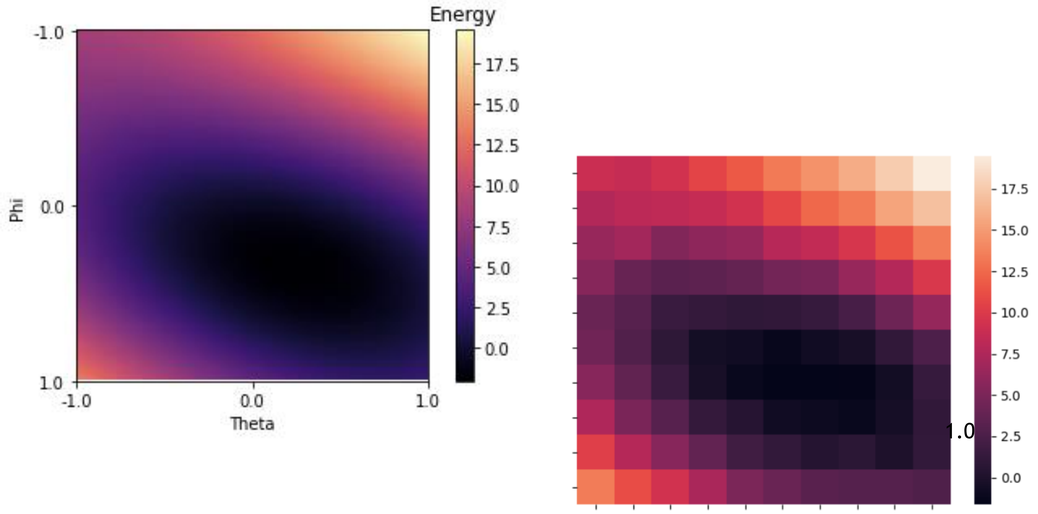
Seamlessly Target any Quantum Resource

CUDA QUANTUM PLATFORM



CUDA Quantum and Quantinuum

```
nvq++ -qpu=cuquantum vqe.cpp
```



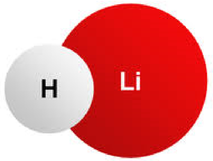

```
nvq++ -qpu=quantinuum:h1 vqe.cpp
```

github.com/nvidia/cuda-quantum | https://catalog.ngc.nvidia.com/orgs/nvidia/containers/cuda-quantum

VQE: NVQ++/cuStateVec on A100 GPU vs Leading Pythonic Framework and SImulator with Thrust on A100 GPU

Novel Configurations with CUDA Quantum

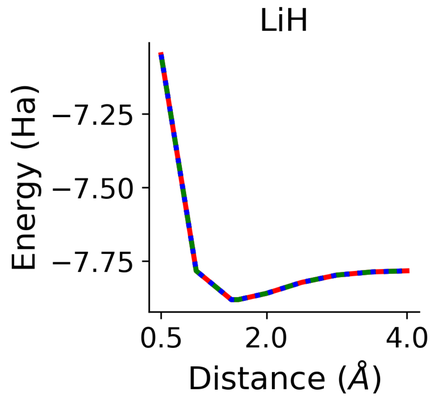
$|\psi(\theta)\rangle$



P_0 P_1 P_2 P_3 ... P_j

P_0 P_1 P_2 P_3 ... P_j

j=11,008 for CO2

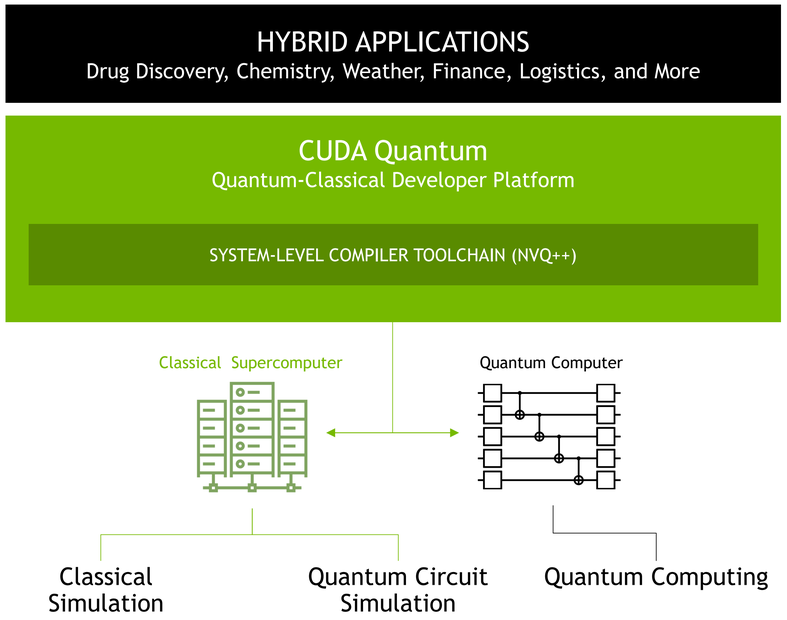


$$\langle\psi(\theta)|H|\psi(\theta)\rangle \approx E_{gs}$$

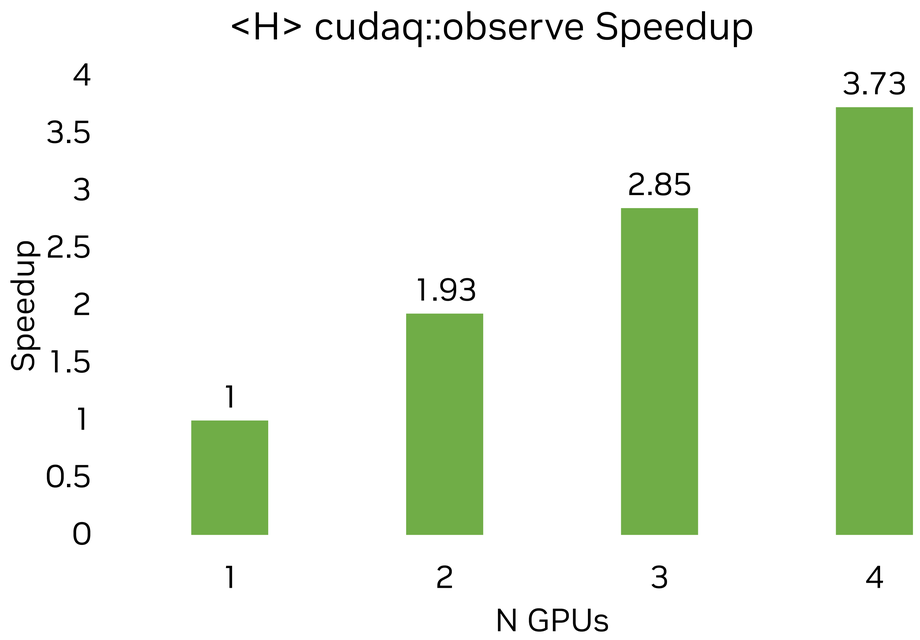
CUDA Quantum: Now Available on GitHub and NGC

Delivering Unmatched Performance, Scalability, And Usability

CUDA QUANTUM PLATFORM



ENGINEERED FOR PERFORMANCE AND SCALE



github.com/nvidia/cuda-quantum | https://catalog.ngc.nvidia.com/orgs/nvidia/containers/cuda-quantum

VQE: NVQ++/cuStateVec on A100 GPU vs Leading Pythonic Framework and SImulator with Thrust on A100 GPU

Thank you