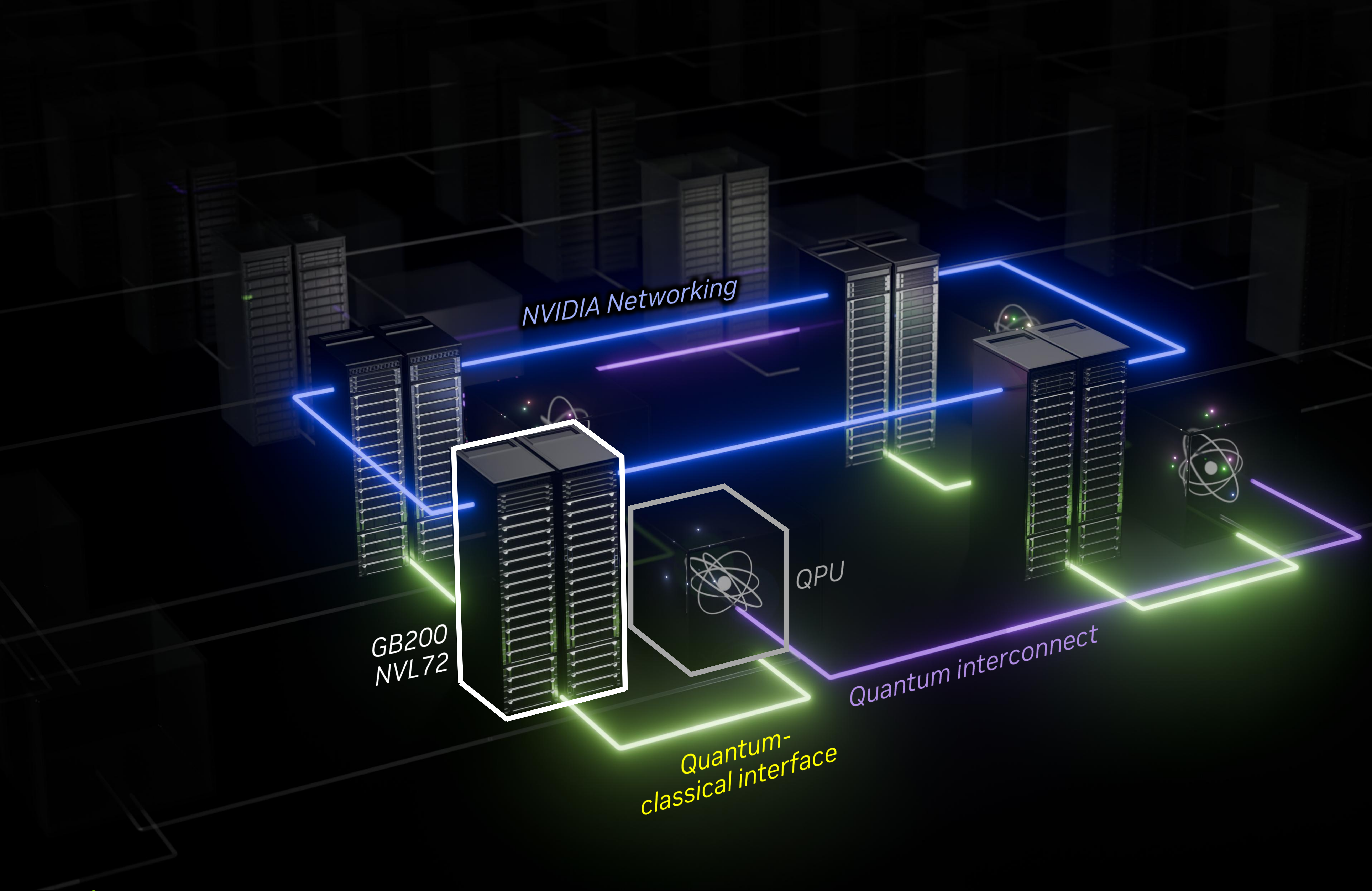
Accelerated Quantum Supercomputing

Yuri Alexeev, Senior Quantum Algorithm Engineer,
NVIDIA Corporation

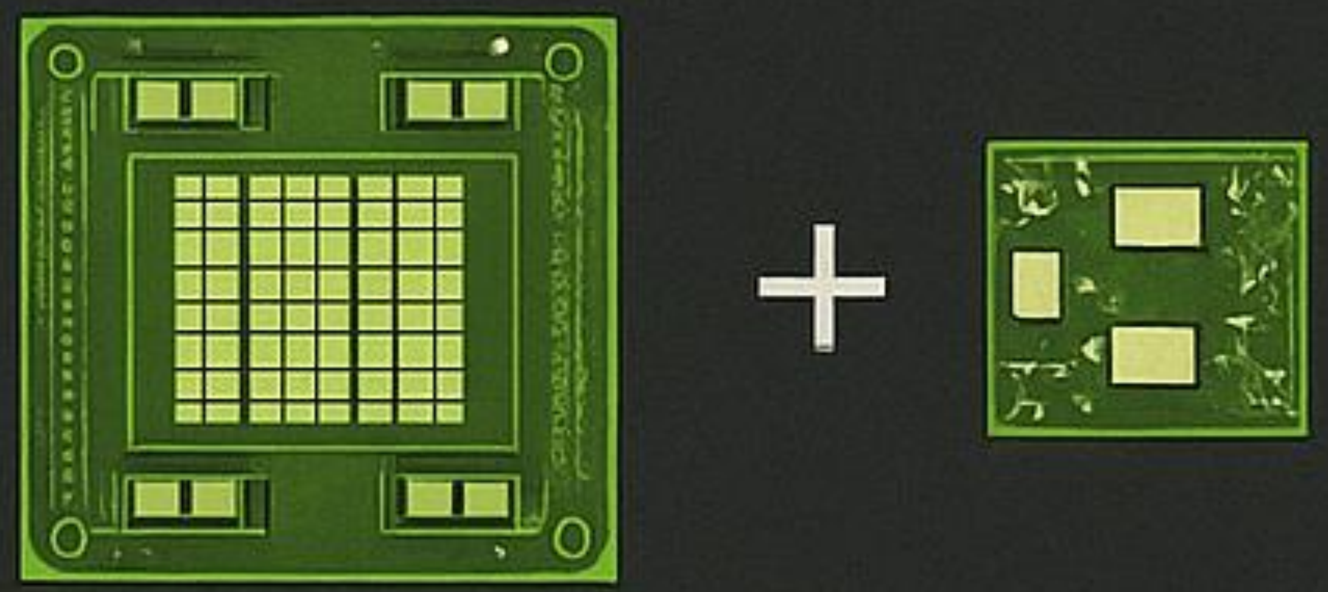
Accelerated Quantum Supercomputing

NVIDIA supercomputers leveraging QPUs



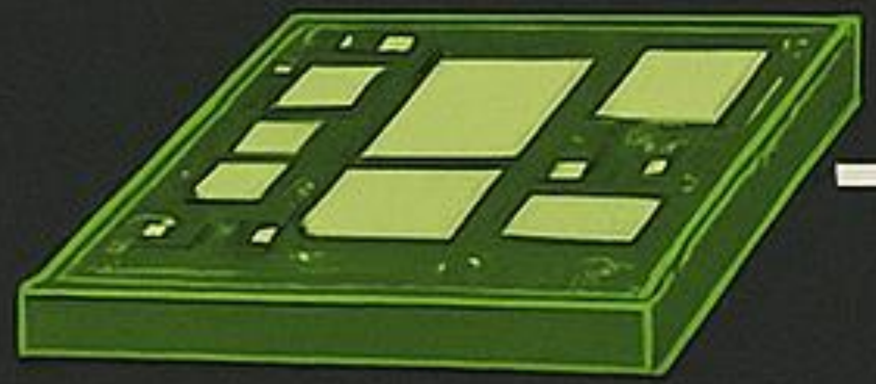
- NVIDIA supercomputers **integrate quantum computers** as a co-processor
- NVIDIA's solutions **de-risk the quantum industry** by being agnostic to the different QPU modalities
- **Hybrid applications** need GPUs and QPUs
- NVIDIA's **CUDA-Q software framework** allows for seamless applications programming

BUILDING BLOCKS



GPU CHIP

Single computational unit (e.g, H100, B200, GB
Includes CUDX cores, fensor cores, and
HBM memory Fundamental unit for AI conpution

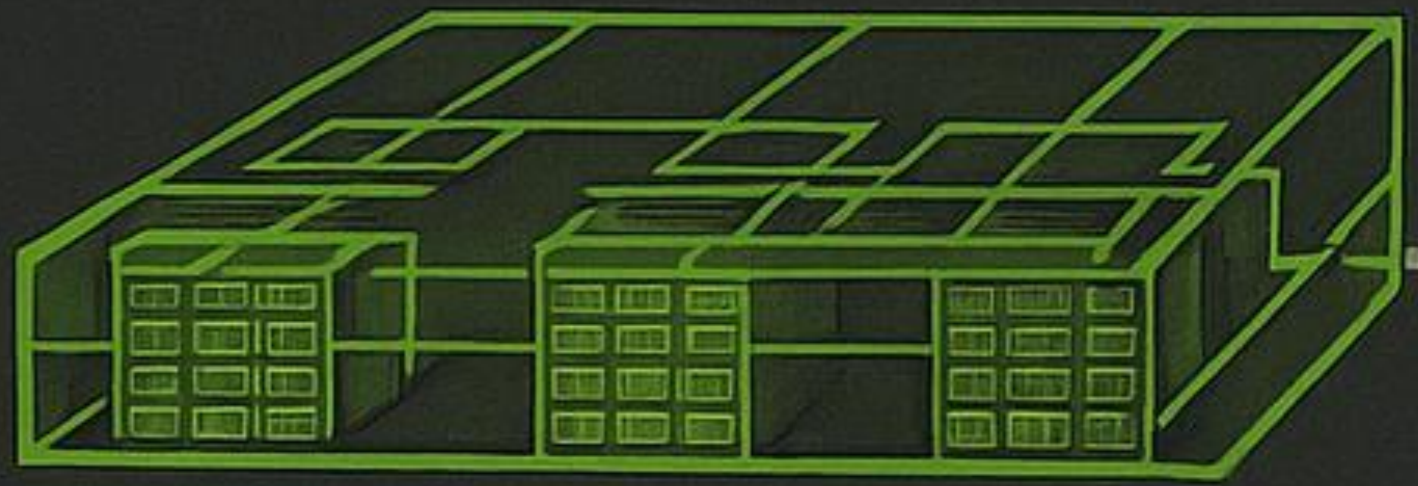


SUPERCHIP

Integrated package: multiple.processors.
Example. GB200 + 2 x B200 + 1 x Grace CPU
High -speed NVLink-C2C (900 GB/s) interconnect

DGX SYSTEM

Purpose-built AI server
Contains 8 GPUs or multiple superchips
Includes networking, storage, and cooling in one chassis



POD / SCALABLE UNIT (SU)

Composed of 32-64 DGX systems
Integrated with high-bandwidth networking and storage
Provides –1,000+ GPUs in synchronized deployment

SUPERPOD

Data center-scale AI intrastructure
Combines 128=2,040+ DGX systems
Delivers exaflop-scale performancerre
Includes-full-stack, power, cooling, orchestral.loftware

Accelerated Quantum Supercomputing

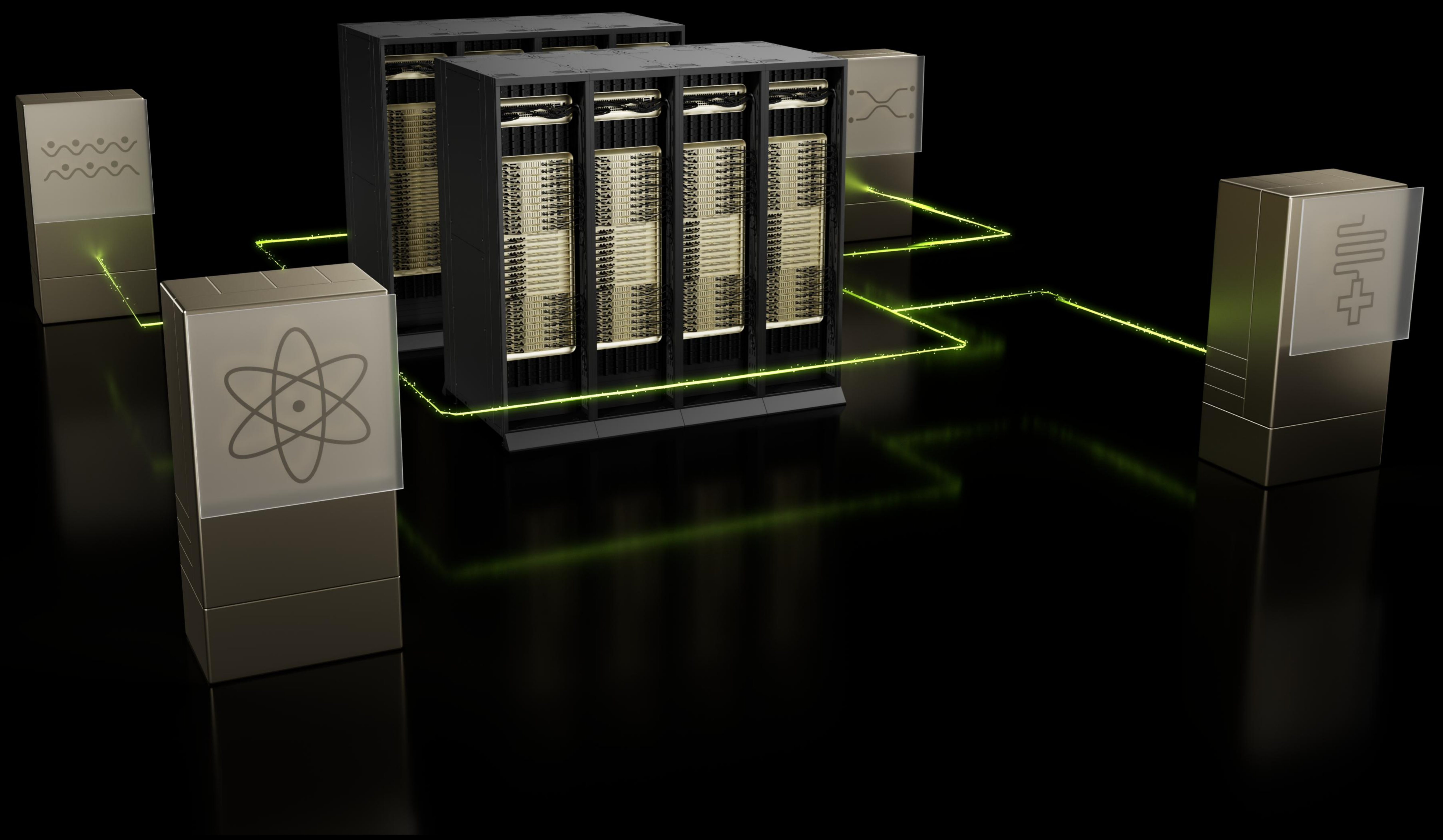
The NVIDIA Accelerated Quantum Research Center (NVAQC)

Boston, Massachusetts

GB200 NVL72 pods

Partner quantum hardware

Research to enable quantum accelerated supercomputing



EQuS ENGINEERING QUANTUM SYSTEMS

Harvard Quantum Initiative IN SCIENCE AND ENGINEERING

QUANTINUUM

QUANTUM MACHINES

|QuEra> Computing Inc.

GB200 NVL72

Configuration	36 Grace CPU : 72 Blackwell GPUs	1 Grace CPU : 2 Blackwell GPU
FP4 Tensor Core[1]	1,440 PFLOPS	40 PFLOPS
FP8/FP6 Tensor Core[1]	720 PFLOPS	20 PFLOPS
INT8 Tensor Core[1]	720 POPS	20 POPS
FP16/BF16 Tensor Core[1]	360 PFLOPS	10 PFLOPS
TF32 Tensor Core	180 PFLOPS	5 PFLOPS
FP32	5,760 TFLOPS	160 TFLOPS
FP64	2,880 TFLOPS	80 TFLOPS
FP64 Tensor Core	2,880 TFLOPS	80 TFLOPS
GPU Memory | Bandwidth	Up to 13.4 TB HBM3e | 576 TB/s	Up to 372GB HBM3e | 16 TB/s
NVLink Bandwidth	130TB/s	3.6TB/s
CPU Core Count	2,592 Arm® Neoverse V2 cores	72 Arm Neoverse V2 cores
CPU Memory | Bandwidth	Up to 17 TB LPDDR5X | Up to 18.4 TB/s	Up to 480GB LPDDR5X | Up to 512 GB/s

GB200 NVL72

Highlights

Supercharging Next-Generation AI and Accelerated Computing

LLM Inference	LLM Training	Energy Efficiency	Data Processing
30X vs. NVIDIA H100 Tensor Core GPU	4X vs. H100	25X vs. H100	18X vs. CPU

The Journey From Qubits to Supercomputers

NVIDIA

DGX Quantum

System for Integration of Quantum with GPU supercomputing

- Tightly integrates Quantum with GPU Supercomputing
- Qubit Agnostic – Supports different qubit modalities
- Reduces GPU-QPU latency by 1-2 orders of magnitude
- Enables GPU Acceleration of Quantum Error Correction, Calibration, and Hybrid Algorithms
- Scalable for more GPU compute and larger QPUs

NVIDIA.

QUANTUM MACHINES

DGX Quantum

System for Integration of Quantum with GPU supercomputing

Typical Latencies

Classical-Quantum Latencies

Remote QPU, Web API	1-10 seconds
Local QPU, Ethernet	10 microseconds
Typical Error Correction Budget*	10 microseconds
DGX Quantum PCIe	400 nanoseconds (PCIe 5.0 100 nanoseconds)
DGX Quantum	4 microseconds

*Includes decoding time

DGX A100 node

NVIDIA Quantum Product Map

CUDA-Q is the entry point into our products for most users

NVIDIA Quantum Cloud

CUDA-Q Academic

CUDA-Q

Libraries | Programming Model | Tools | Infrastructure

Quantum

Classical

cuQuantum

Libraries for simulating applications and hardware

QPU

CUDA-Q supports range of QPU vendor backends

OQC, Quantinuum, ORCA Computing, Alice & Bob, IONQ, Quantum Brilliance, Pasqal, IQM, QuEra Computing Inc., seeqc, rigetti, Infleqtion, Anyon Technologies

DGX-Quantum

Reference architecture for low-latency QPU/DGX integration

DGX-SuperPOD

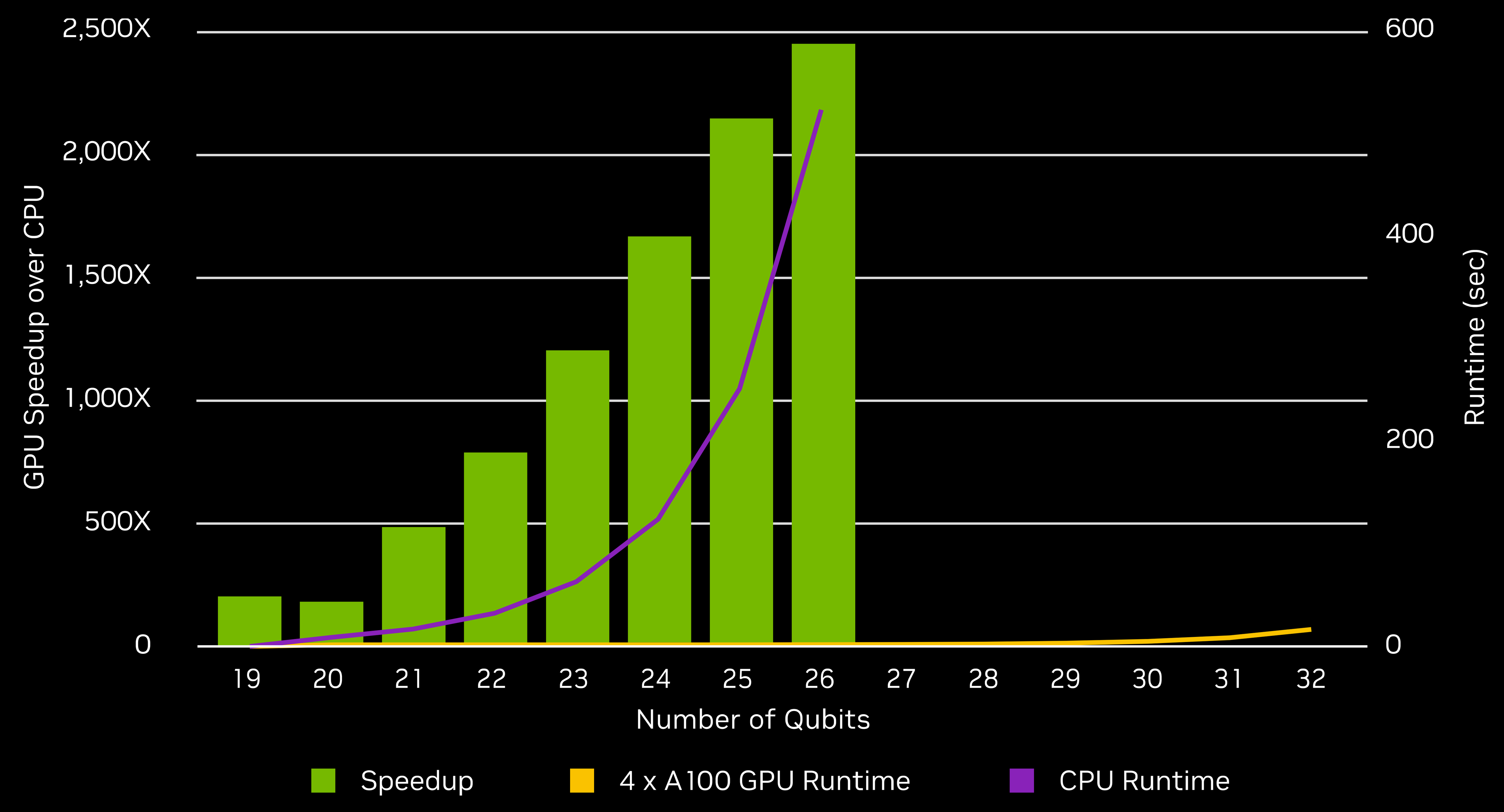
CUDA-Q

The platform for accelerated quantum computing

Features

- **Python and C++**
  - Access via familiar & powerful languages
- **QPU agnostic**
  - Optimized backends from all major QPU vendors and qubit modalities
- **GPU-accelerated** simulation
  - Quantum simulators that scale to large-scale quantum computers
- Fully **kernel system for hybrid computing interface**
  - Seamlessly combine GPU and QPU resources
- Supports **QEC HW development**
  - DGX-Quantum reference architecture allows decoder and calibration development
- Access to classical **CUDA-X and AI libraries**
  - Conventional parts of hybrid algorithms can draw on fastest implementations
- Comprehensive **educational tools**
  - CUDA-Q Academic onboards users to accelerated quantum supercomputing

Performance



QML workflow in CUDA-Q using multithreaded CPU versus NVIDIA A100 Tensor Core GPUs

Getting started with CUDA-Q

CUDA-Q Overview
https://developer.nvidia.com/cuda-q

CUDA-Q Academic
https://github.com/NVIDIA/cuda-q-academic

CUDA-Q Docs
https://nvidia.github.io/cuda-quantum/latest/index.html

CUDA-Q Apps
https://nvidia.github.io/cuda-quantum/latest/using/tutorials.html

Defining the Accelerated Quantum Supercomputer

A New Heterogenous Architecture

Programming model and compiler for heterogenous supercomputer

Used for programming low-latency real-time hybrid applications

Libraries to enable domain scientists

Open source and qubit-agnostic

C++

- Kernel Expressions
- JIT Kernel Expressions
- Runtime

Python

- Kernel Expressions
- JIT Kernel Expressions
- Runtime

CUDA-Q Intermediate Representation (MLIR)

Quake	CC	Func	Math	Arith	LLVM
(Quantum)	(Classical CFG)	(Kernels)	(Standard Math)	(Constants)	(Lowering Target)

Quantum Intermediate Representation (QIR, Profiles, LLVM IR)

Simulation (MGPU, MNMG, DM, TN)

Physical QPU (Quantinuum, IonQ, IQM, OQC...)

Role of IRs in CUDA-Q

Purpose	Role of IRs
Abstraction	Separates front-end languages (C++, Python) from backend targets (simulators/QPU).
Optimization	Enables compiler-level transformations of quantum and classical code.
Target Independence	Facilitates code generation for simulators (MGPU, MPS, TN) and QPUs.
Modularity and Composability	Supports analysis, transformation, and instrumentation at multiple levels.
AI/ML Integration	Allows insertion of AI-driven rewrites or cost model heuristics via IR pass.

GHZ State Example

Running on GPU

```python
import cudaq

@cudaq.kernel
def ghz_state(N: int):
    qubits = cudaq.qvector(N)
    h(qubits[0])
    for i in range(N - 1):
        x.ctrl(qubits[i], qubits[i + 1])
    mz(qubits)
```

```python
cudaq.set_target(“nvidia")
n = 29
print("Preparing GHZ state for", n, "qubits.")
counts = cudaq.sample(ghz_state,n)
counts.dump()
```

Output:

```
Preparing GHZ state for 29 qubits.
{ 00000000000000000000000000000:509 11111111111111111111111111111:491 }
```

Challenges facing HCP-quantum integration

- \- **Hardware challenges** concern the design of tightly coupled HPC–quantum systems. Co-locating quantum and classical resources within the same hardware node is essential for the low-latency communication and tight synchronization required.
- \- **Software challenges** involve creating a unified, seamless software stack enabling the efficient orchestration of quantum and classical components.
- \- **Algorithmic challenges** lie in developing quantum algorithms designed for hybrid HPC–FTQC platforms. There is a significant gap in algorithms tailored to the intermediate regime, where a small number of logical qubits coexist with HPC. Such algorithms must leverage distributed quantum and classical resources and may require novel co-design approaches, potentially leveraging the use of AI.

Quantum Computing Needs AI Supercomputing

Quantum Development

Algorithms and applications research

QPU design

QEC research

Training AI models for:
-QEC
-Control
-Calibration

Quantum Deployment

Real-time accelerated QEC

AI-assisted calibration, control, and readout

Hybrid algorithms and applications

Artificial Intelligence for Quantum Computing

Yuri Alexeev†[1], Marwa H. Farag†[1], Taylor L. Patti†[1], Mark E. Wolf†[1]*, Natalia Ares[2], Alán Aspuru-Guzik[3,4], Simon C. Benjamin[5,6], Zhenyu Cai[5,6], Zohim Chandani[1], Federico Fedele[2], Nicholas Harrigan[1], Jin-Sung Kim[1], Elica Kyoseva[1], Justin G. Lietz[1], Tom Lubowe[1], Alexander McCaskey[1], Roger G. Melko[7,8], Kouhei Nakaji[1], Alberto Peruzzo[9], Sam Stanwyck[1], Norm M. Tubman[10], Hanrui Wang[11] and Timothy Costa[1]

[1]NVIDIA Corporation, 2788 San Tomas Expressway, Santa Clara, 95051, CA, USA.

[2]Department of Engineering Science, University of Oxford, Parks Road, Oxford, OX1 3PJ, United Kingdom.

[3]Department of Chemistry, Computer Science, Materials Science and Engineering, and Chemical Engineering and Applied Science,University of Toronto, 80 St George St, Toronto, M5S 3H6, ON, Canada.

[4]Vector Institute for Artificial Intelligence, 661 University Ave Suite 710, Toronto, M5G 1M1, ON, Canada.

[5]Quantum Motion *, 9 Sterling Way, London, N7 9HJ, United Kingdom.

[6]Department of Materials, University of Oxford, Parks Road, Oxford, OX1 3PH, United Kingdom.

[7]Department of Physics and Astronomy, University of Waterloo, 200 University Avenue West., Waterloo, N2L 3G1, ON, Canada.

[8]Perimeter Institute for Theoretical Physics, 31 Caroline Street North, Waterloo, N2L 2Y5, ON, Canada.

[9]Qubit Pharmaceuticals, 29, rue du Faubourg Saint Jacques, Paris, 75014, France.

[10]NASA Ames Research Center, Moffett Field, California, 94035-1000, USA.

arXiv:2411.09131v1

AI to Enable Quantum Computing

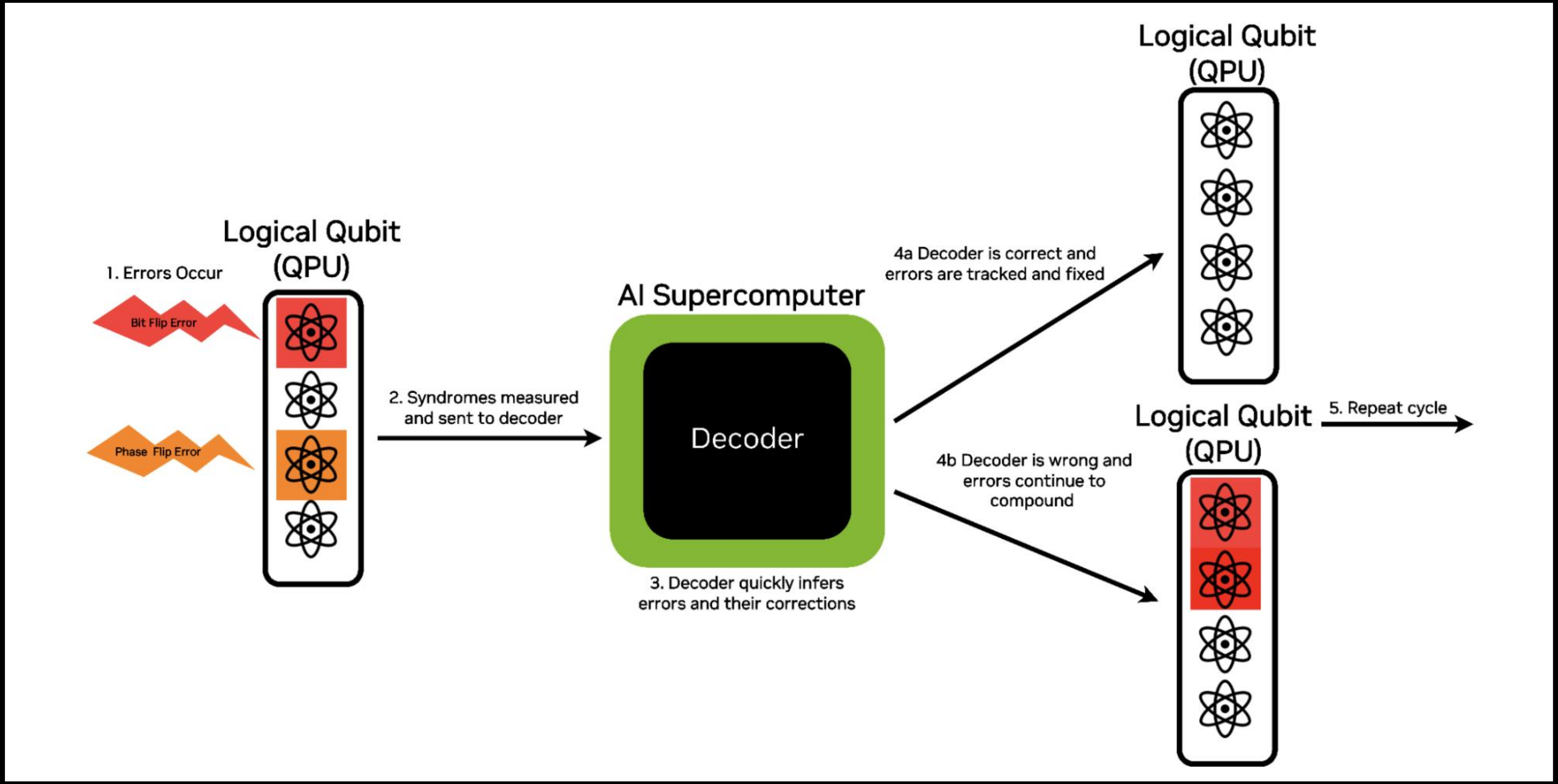
NVIDIA

Scaling Quantum Error Correction: A Critical Challenge

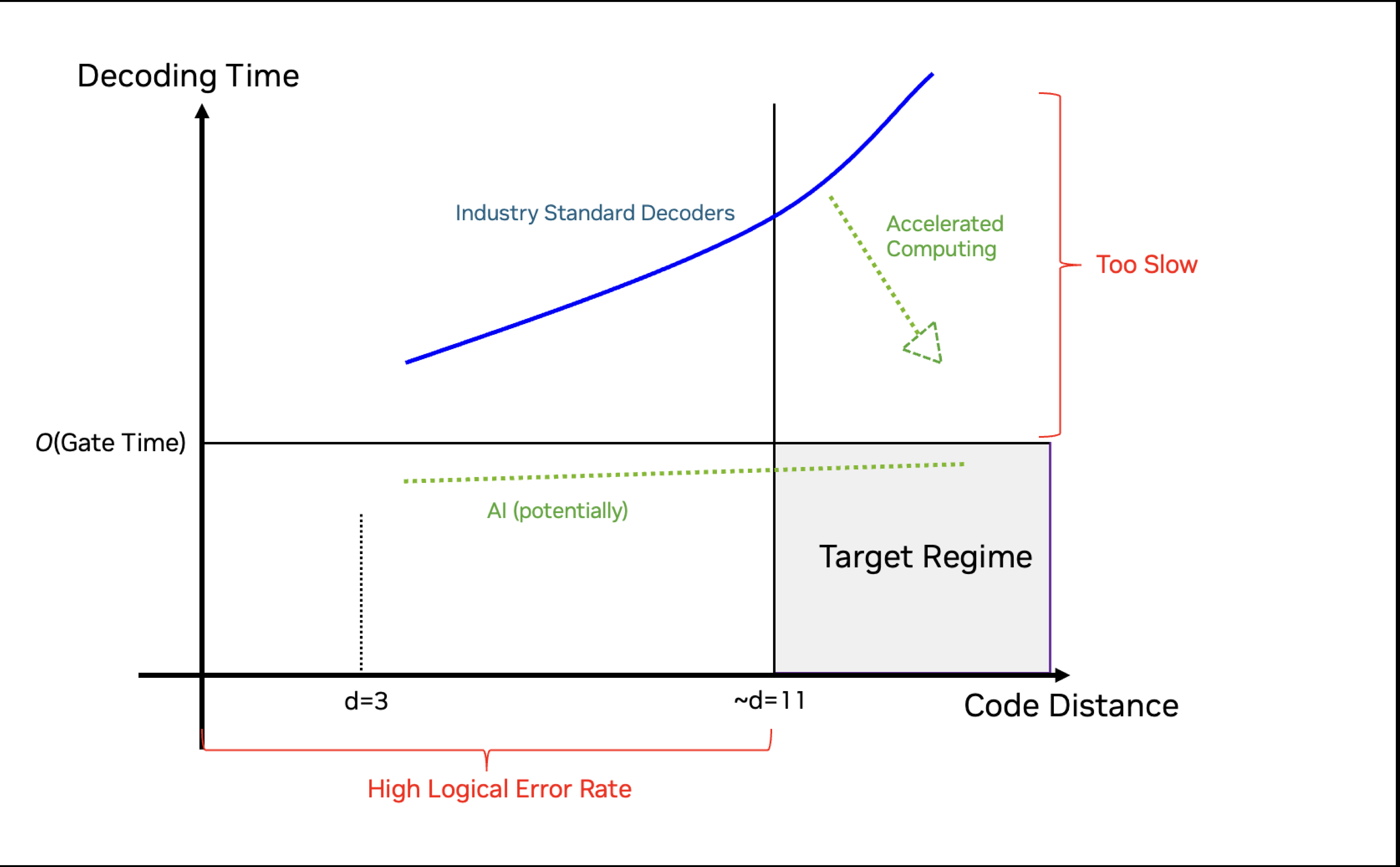
Fault-Tolerant QC is mostly QEC

10^{-3}
State of the art error rates

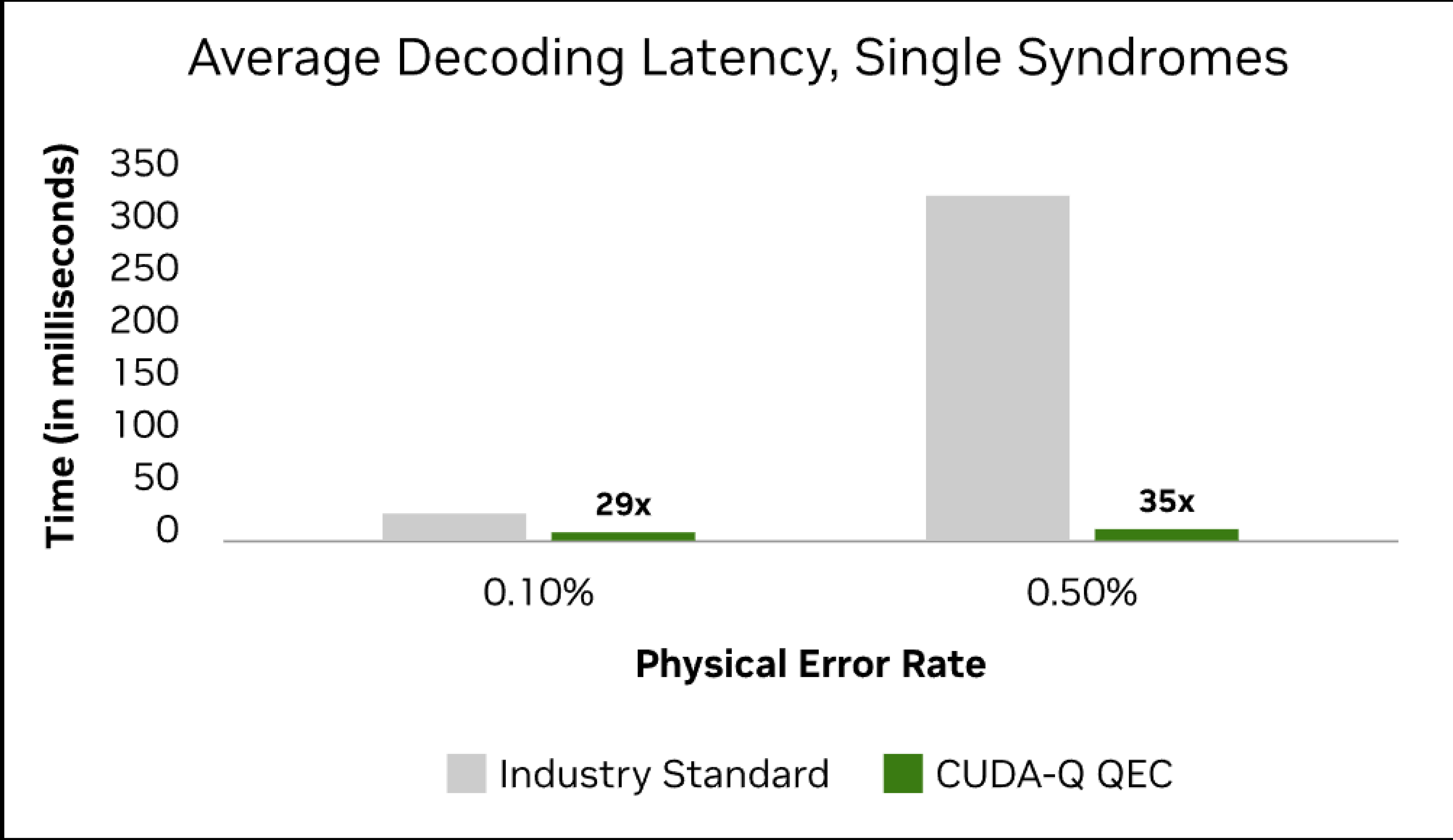
$<10^{-10}$
Expected Error rates needed



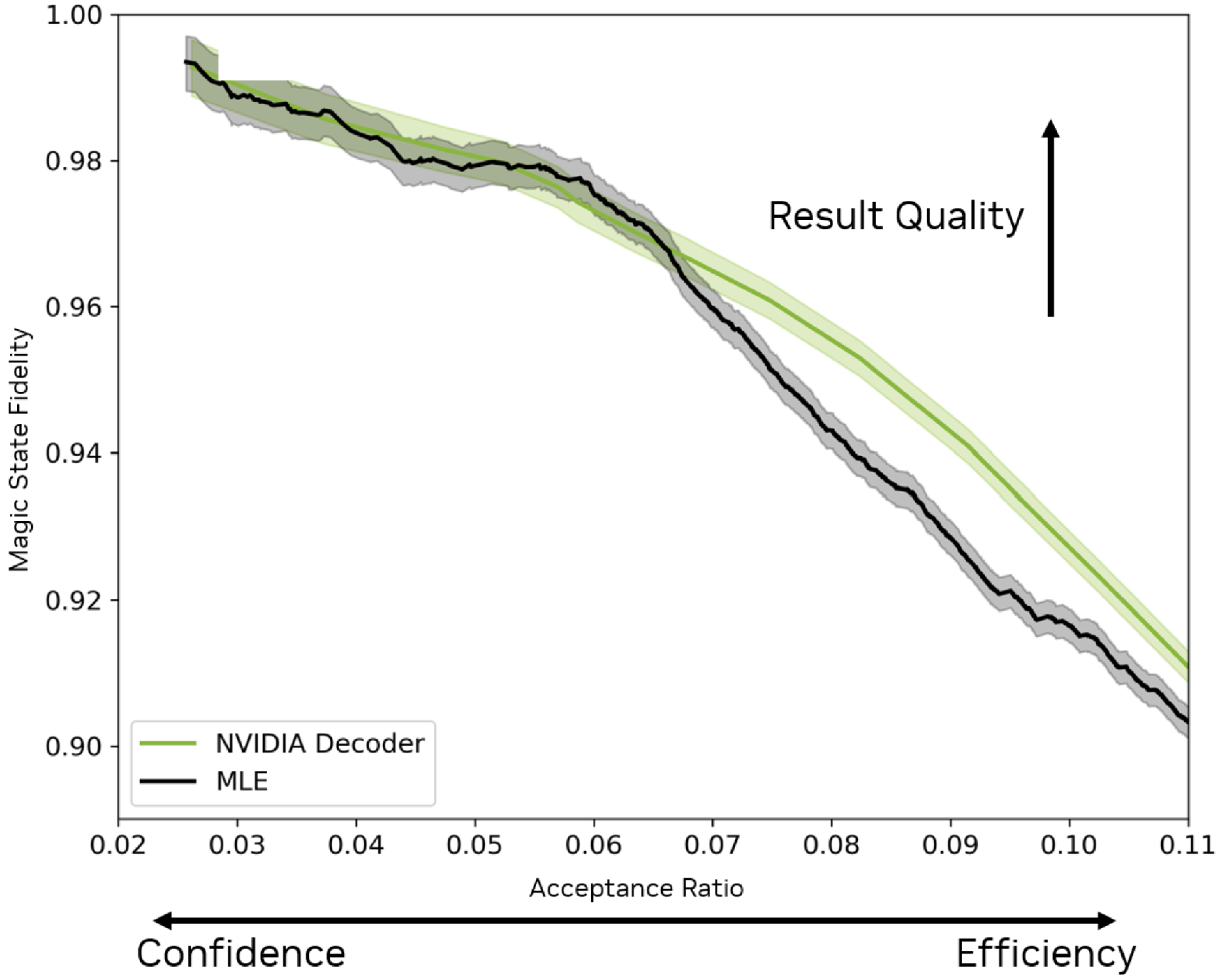
Scaling Quantum Error Correction: A Critical Challenge

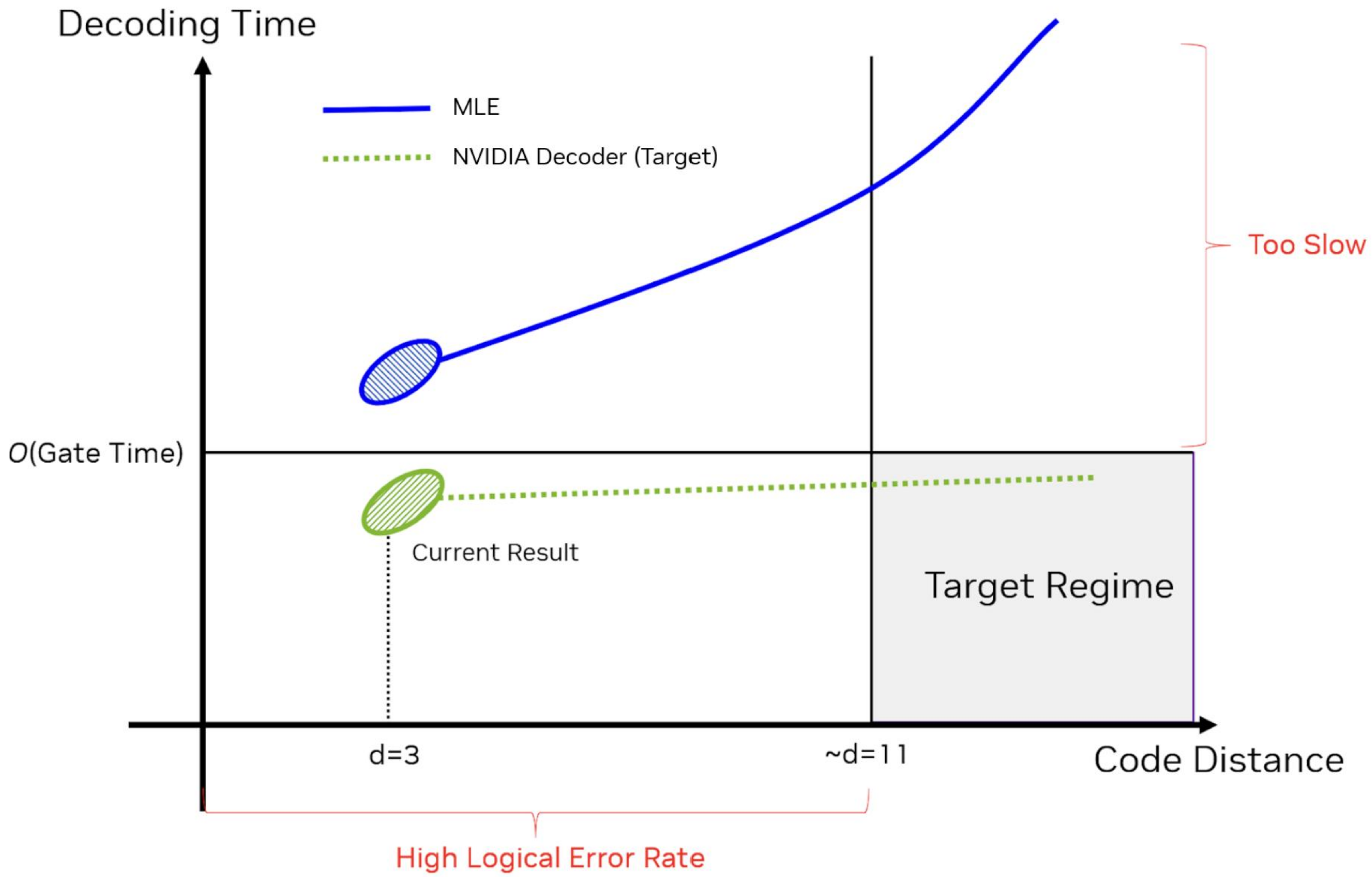


Scaling Quantum Error Correction: A Critical Challenge



AI Decoding Outperforms MLE





Challenges of VQE and ADAPT-VQE algorithms

- Slow Convergence in Plateau Regions:

  leading to very slow convergence

- Number of Gradient Evaluations:

  high measurement overhead . Not scalable. practical applicability rapidly diminishes with system scaling

- Operator Pool Size and Completeness:

  the choice of the operator pool directly impacts efficiency, accuracy, and convergence.

- Number of measurements:

  $O\left(\frac{1}{\epsilon^2}\right)$ with an additive error ϵ (ϵ determines the precision of the result)

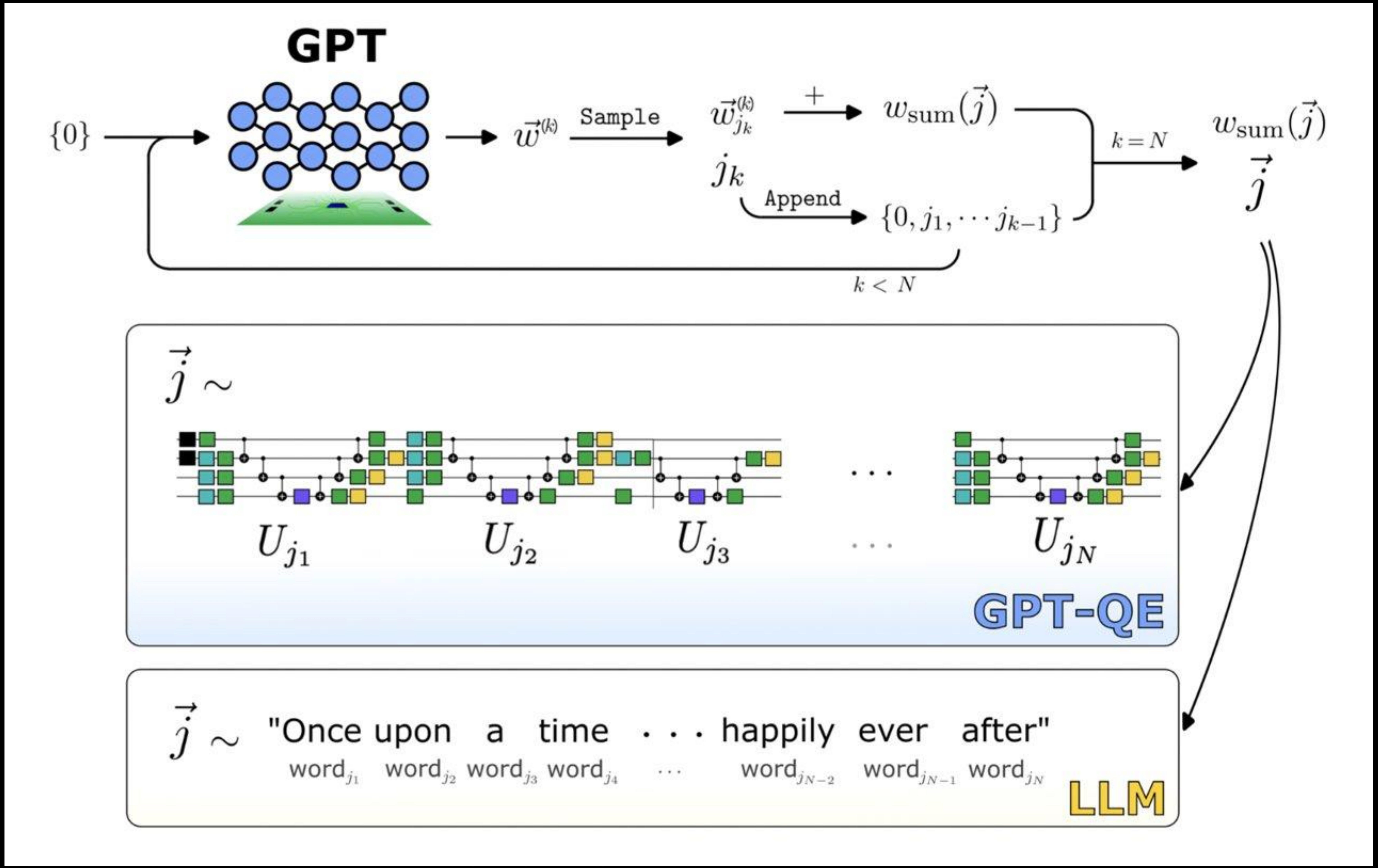
Can generative AI be a promising route?

The generative quantum eigensolver (GQE) and its application for ground state search

Kouhei Nakaji[1,2,3], Lasse Bjørn Kristensen[1,4], Jorge A. Campos-Gonzalez-Angulo *[1], Mohammad Ghazi Vakili *[1,4], Haozhe Huang *[4,5], Mohsen Bagherimehrab †[1,4], Christoph Gorgulla †[6,7], FuTe Wong[4,8], Alex McCaskey[9], Jin-Sung Kim[9], Thien Nguyen[9], Pooja Rao[9], and Alan Aspuru-Guzik[1,4,5,10,11,12]

[1] Chemical Physics Theory Group, Department of Chemistry, University of Toronto, Toronto, Ontario, Canada
[2] Research Center for Emerging Computing Technologies, National Institute of Advanced Industrial Science and Technology (AIST), 1-1-1 Umezono, Tsukuba, Ibaraki, Japan
[3] Quantum Computing Center, Keio University, 3-14-1 Hiyoshi, Kohoku-ku, Yokohama, Kanagawa, Japan
[4] Department of Computer Science, University of Toronto, Toronto, Ontario, Canada
[5] Vector Institute for Artificial Intelligence, Toronto, Ontario, Canada
[6] Department of Physics, Faculty of Arts and Sciences, Harvard University, Cambridge, Massachusetts, USA
[7] Department of Structural Biology, St. Jude Children's Research Hospital, Memphis, Tennessee, USA
[8] Institute of Medical Science, University of Toronto, Toronto, Ontario, Canada
[9] NVIDIA, Santa Clara, California, USA
[10] Department of Chemical Engineering & Applied Chemistry, University of Toronto, Toronto, Ontario, Canada
[11] Department of Materials Science & Engineering, University of Toronto, Toronto, Ontario, Canada
[12] Lebovic Fellow, Canadian Institute for Advanced Research, Toronto, Ontario, Canada

Generative Quantum Eigensolver (GPT-QE)



Probability that a sequence of ***j*** is sampled is determined by the logits sum:

$\boldsymbol{j} \sim e^{-\beta W(\boldsymbol{j})}$ $\quad W(\boldsymbol{j}) = W_{j1} + W_{j2} + \ldots + W_{jN}$

If W(j) = E(j) and β is large, the ground state is likely to be generated

Logit matching:

Cost= $(W(\boldsymbol{j}) - E(\boldsymbol{j}))^2$

https://arxiv.org/pdf/2401.09253.pdf

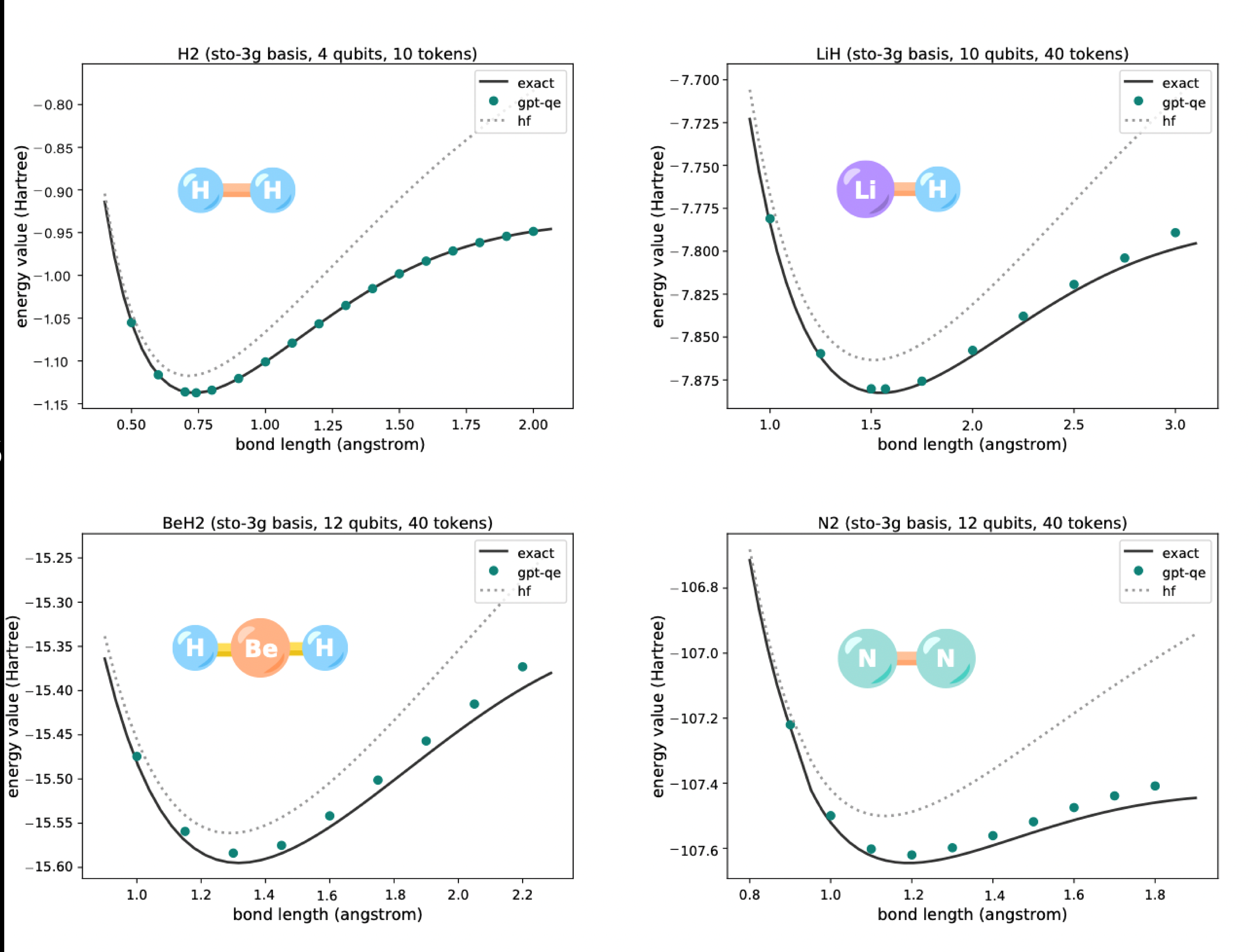
- Quantum gates are analog to words (tokens). Token space includes: gate type, target qubit, evolution time
- Quantum circuits are analog to predicted sentence

UNIVERSITY OF TORONTO St. Jude Children's Research Hospital Finding cures. Saving children. NVIDIA

GPT-QE Performance and Accuracy

Comparing VQE, ADAPT-VQE, GPT-QE

- The first demonstration of a GPT-generated quantum circuit in the literature
- A powerful example of leveraging AI to accelerate quantum computing
- Executed using CUDA Quantum on A100 GPUs on Perlmutter
- Opens the door to a wide variety of novel Generative Quantum Algorithms (GQAs) for drug discovery, materials science, and environmental challenges



- Energies are not within chemical accuracy
- Improving the GPT-QE: work in progress

UNIVERSITY OF TORONTO

St. Jude Children's Research Hospital
Finding cures. Saving children.

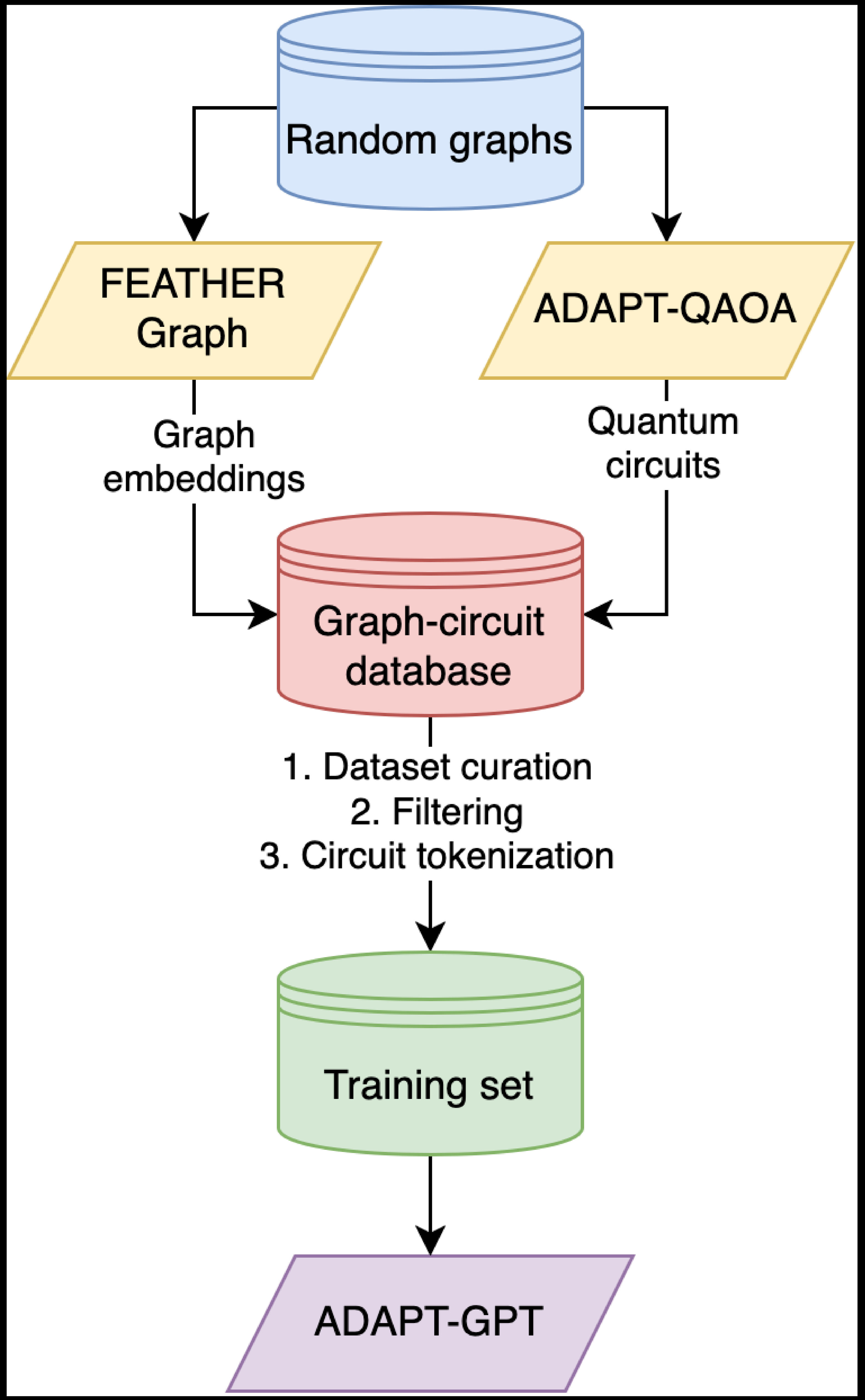
NVIDIA

How to fix GQE

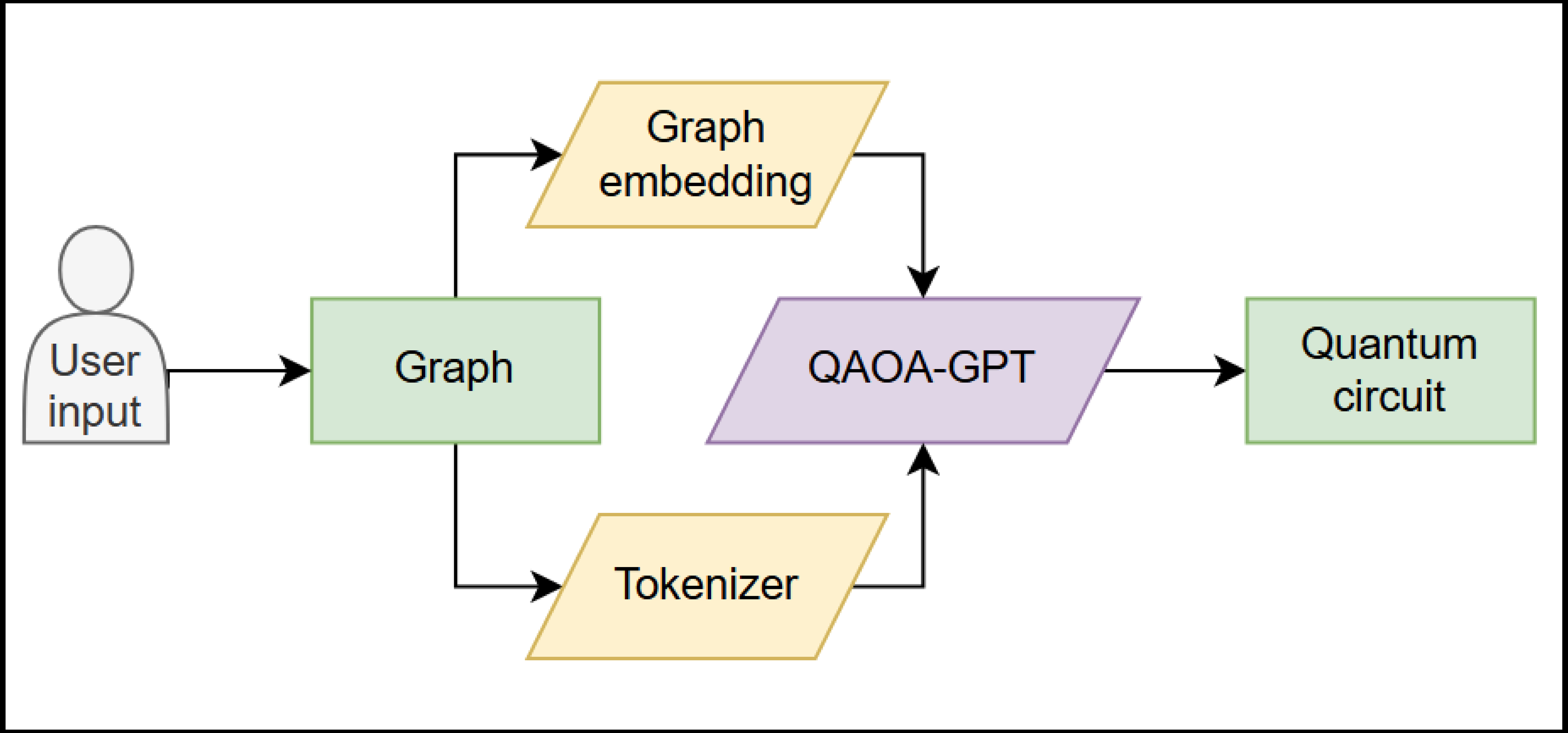
- \- Add energy minimization in cost function (logit matching)
- \- Add quantum concepts to the attention mechanism
- \- Use physics aware neural networks
- \- Train on already optimized quantum circuits

ADAPT-GPT for predicting compact quantum circuit

- Use ADAPT algorithm to generate synthetic data (compact quantum circuits)
- Tokenize the circuit
- The tokenized circuit is then passed to the transformer model for training
- This is called ADAPT-GPT
- ADAPT-GPT can be used to predict compact quantum circuit for other problem not seen before in the training.



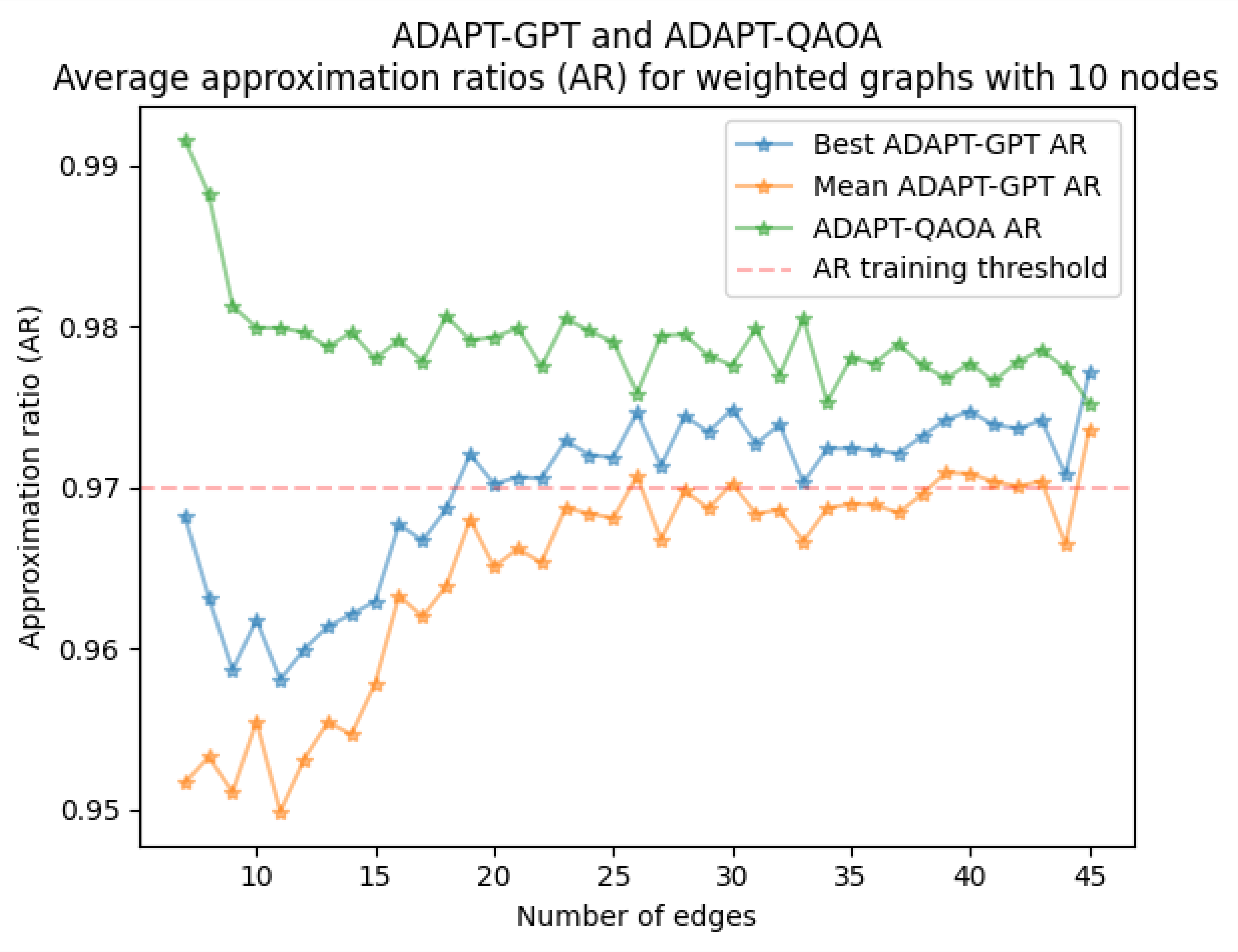
ADAPT-GPT for predicting compact quantum circuit

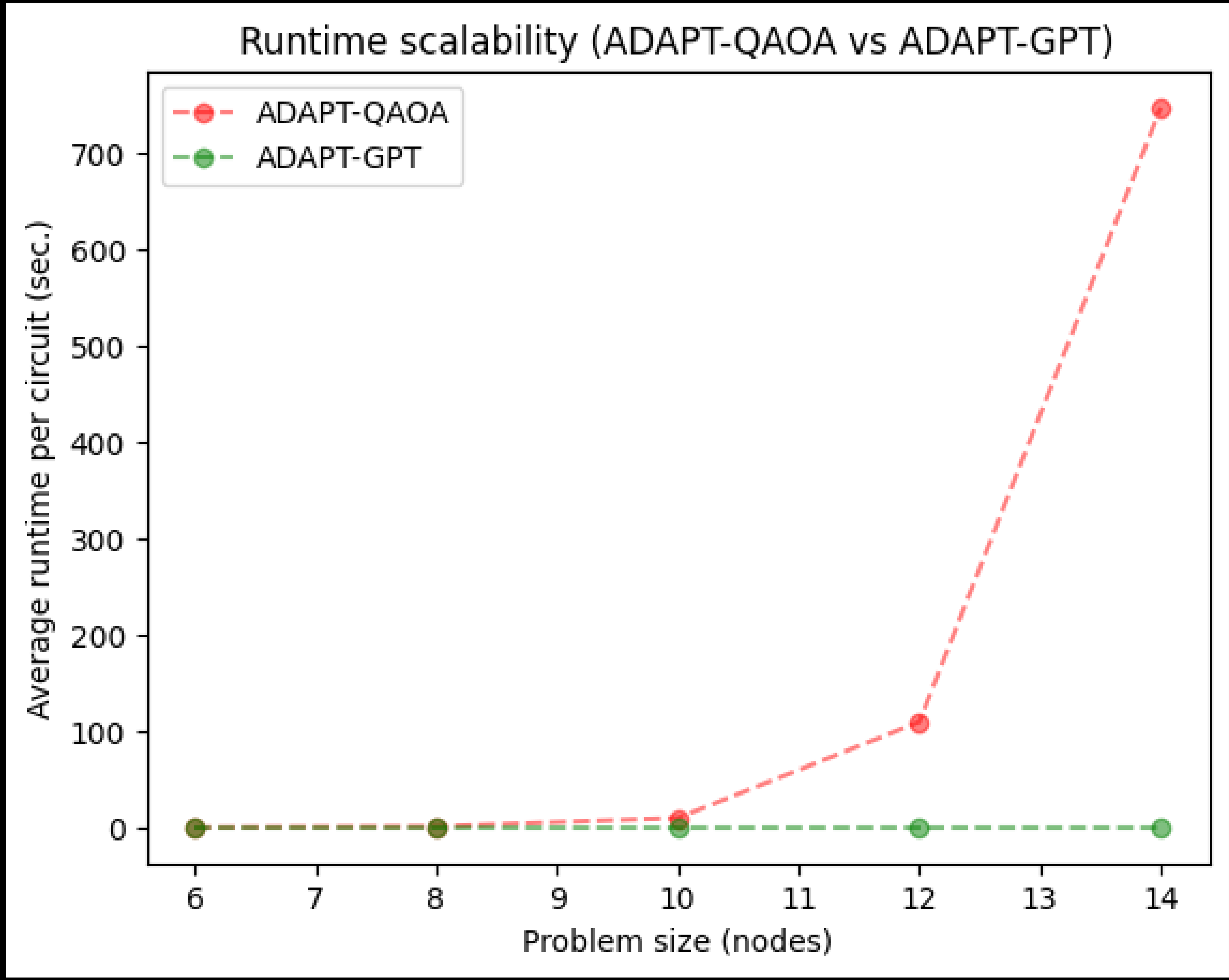


Proposed use case diagram. Given a user-supplied input graph, the system computes a fixed-length graph embedding and tokenizes the graph structure.
Both representations are passed to the QAOA-GPT model, which autoregressively generates a quantum circuit that solves the corresponding QAOA optimization problem.

ADAPT-GPT versus ADAPT-QAOA

Performance and Runtime





Paper accepted to QCE25 conference proceedings

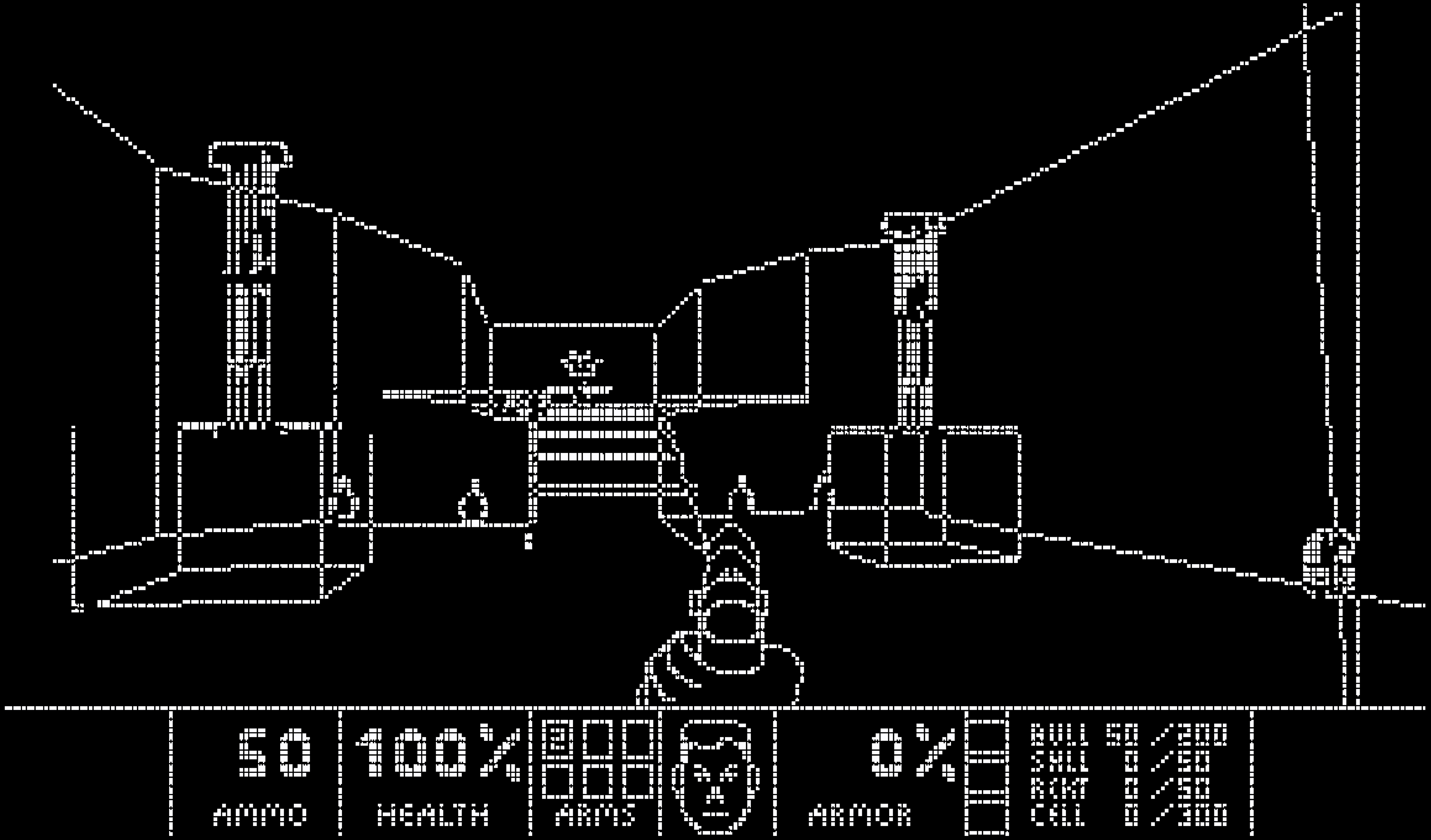
Find out more

NVIDIA Quantum
https://www.nvidia.com/en-us/solutions/quantum-computing/

CUDA-Q v0.12 Now Available
Python – `> pip install cudaq`
C++ – https://github.com/NVIDIA/cuda-quantum/releases

CUDA-QX – QEC and Solvers Libraries
https://developer.nvidia.com/cuda-qx

Quandoom



A port of the first level of DOOM designed for a quantum computer, given as a single QASM file, using a mere 70,000 qubits and 80 million gates. Although such a quantum computer doesn't exist right now, Quandoom is efficiently simulatable on a classical computer, capable of running at 10-20 fps on a laptop

https://github.com/Lumorti/Quandoom

https://arxiv.org/abs/2412.12162v1

NVIDIA

Thank you!

References

- "How to Build a Quantum Supercomputer: Scaling from Hundreds to Millions of Qubits" https://arxiv.org/abs/2411.10406

- "Artificial Intelligence for Quantum Computing" https://arxiv.org/abs/2411.09131

- "The generative quantum eigensolver (GQE) and its application for ground state search" https://arxiv.org/abs/2401.09253