Opportunities and Challenges in Intermediate-Scale Quantum Computing

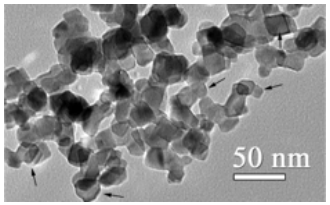
Fred Chong
Seymour Goodman Professor
Department of Computer Science
University of Chicago

Lead PI, the EPiQC Project,
an NSF Expedition in Computing

With Margaret Martonosi, Ken Brown, Peter Shor, Eddie Farhi, Aram Harrow, Diana Franklin, David Schuster, John Reppy, and Danielle Harlow (UChicago, MIT, Princeton, Duke, UCSB)

Why Quantum Computing?

- Fundamentally change what is computable
  - The only means to potentially scale computation exponentially with the number of devices
- Solve currently intractable problems in chemistry, simulation, and optimization
  - Could lead to new nanoscale materials, better photovoltaics, better nitrogen fixation, and more



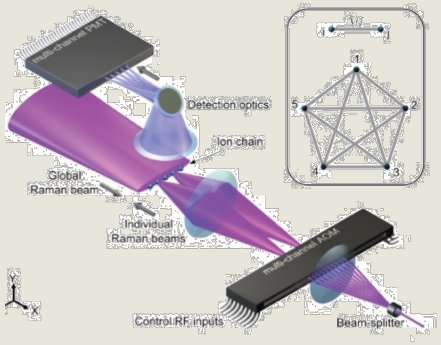

- A new industry and scaling curve to accelerate key applications
  - Not a full replacement for Moore's Law, but perhaps helps in key domains
- Lead to more insights in classical computing
  - Previous insights in chemistry, physics and cryptography
  - Challenge classical algorithms to compete w/ quantum algorithms

NISQ

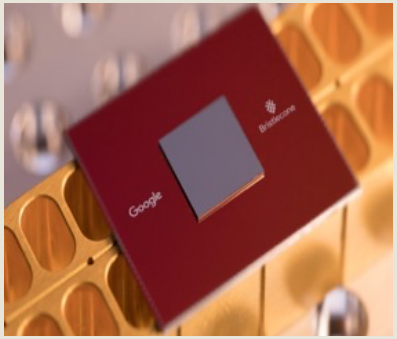
Now is a privileged time in the history of science and technology, as we are witnessing the opening of the NISQ era (where NISQ = noisy intermediate-scale quantum).

– John Preskill, Caltech

IBM Experience
50 superconductor qubits

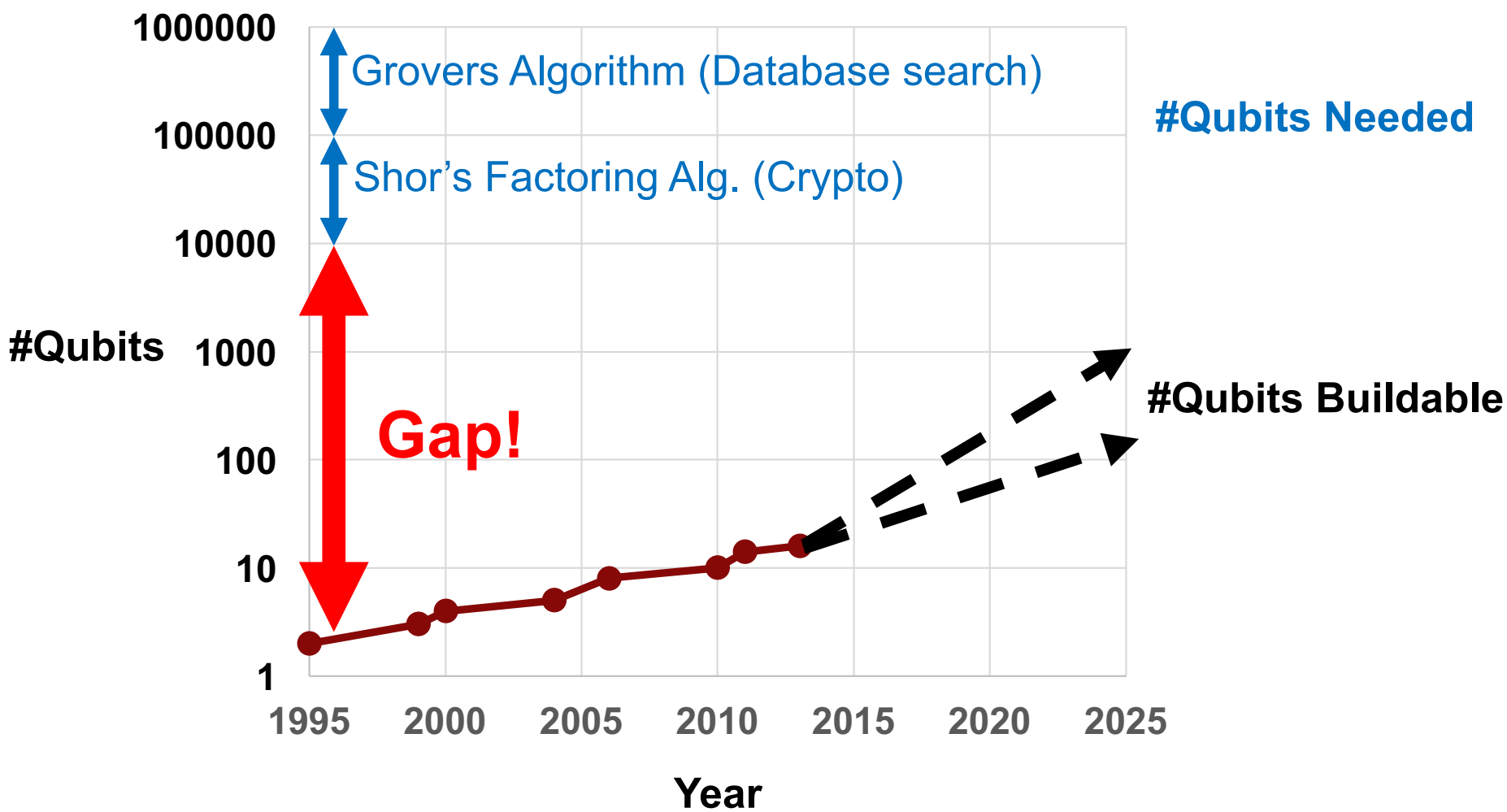


UMD/JQI
9 atomic ion qubits

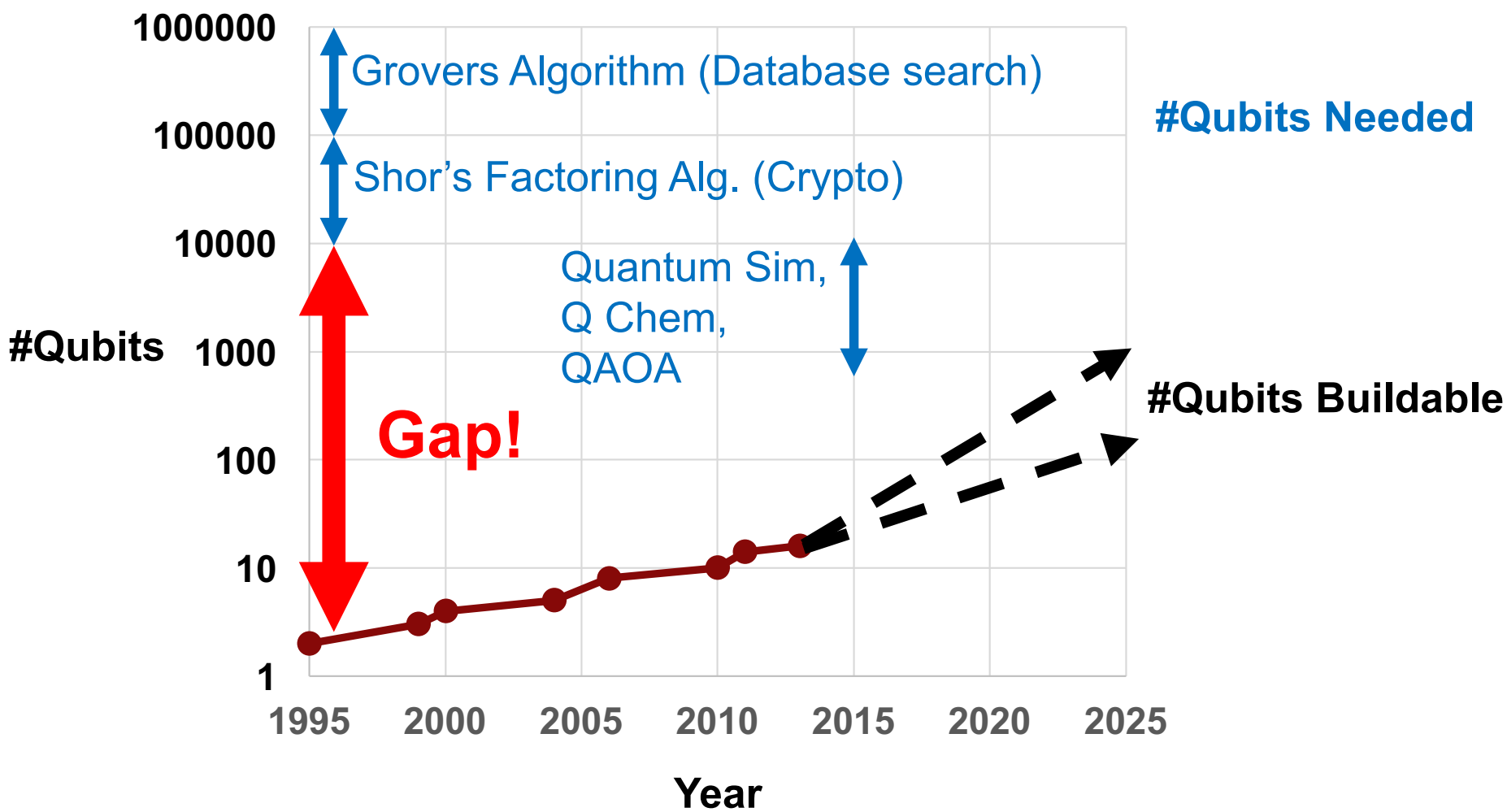


Google
72 supercond qubits

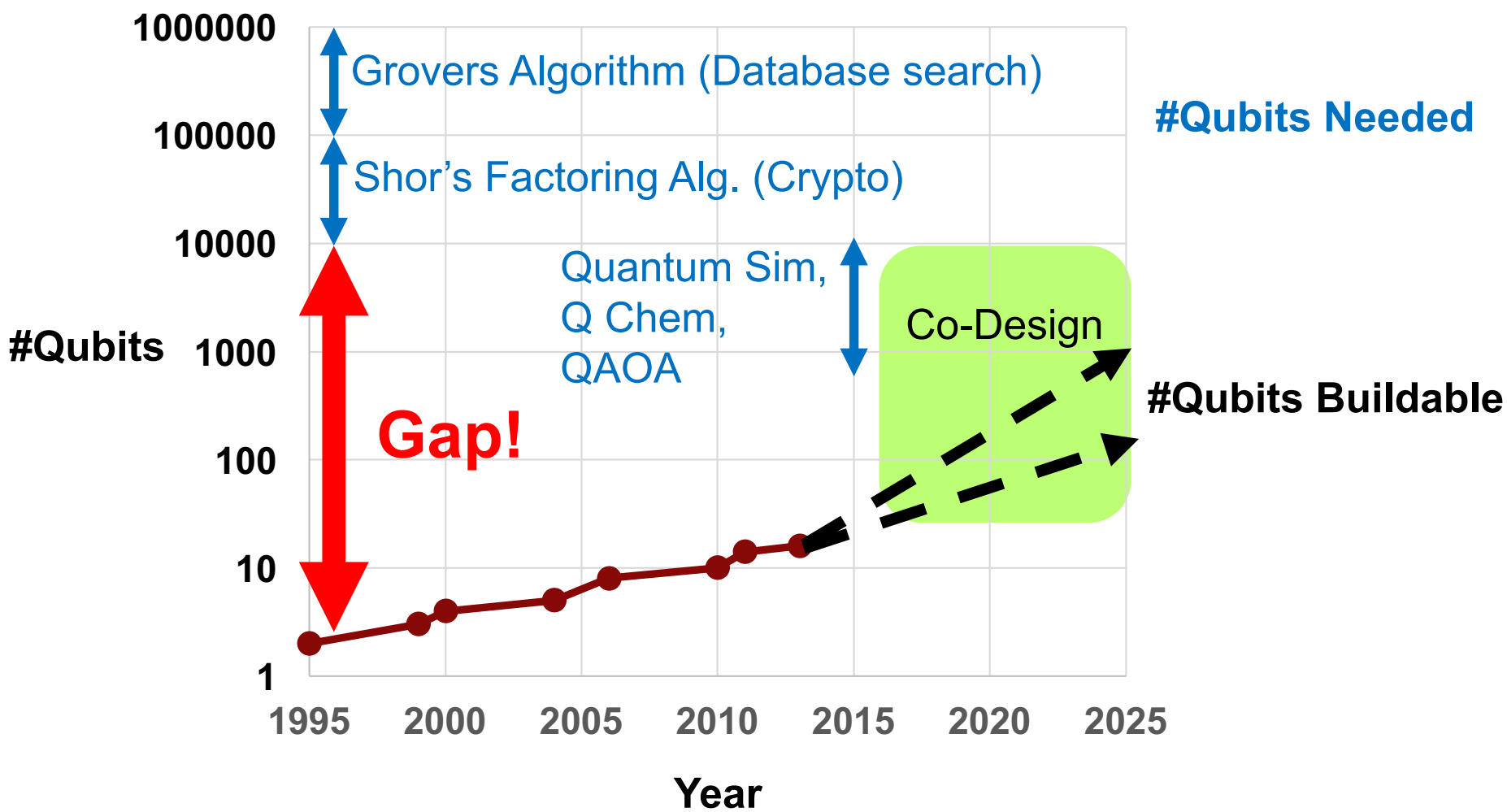
The Algorithms to Machines Gap



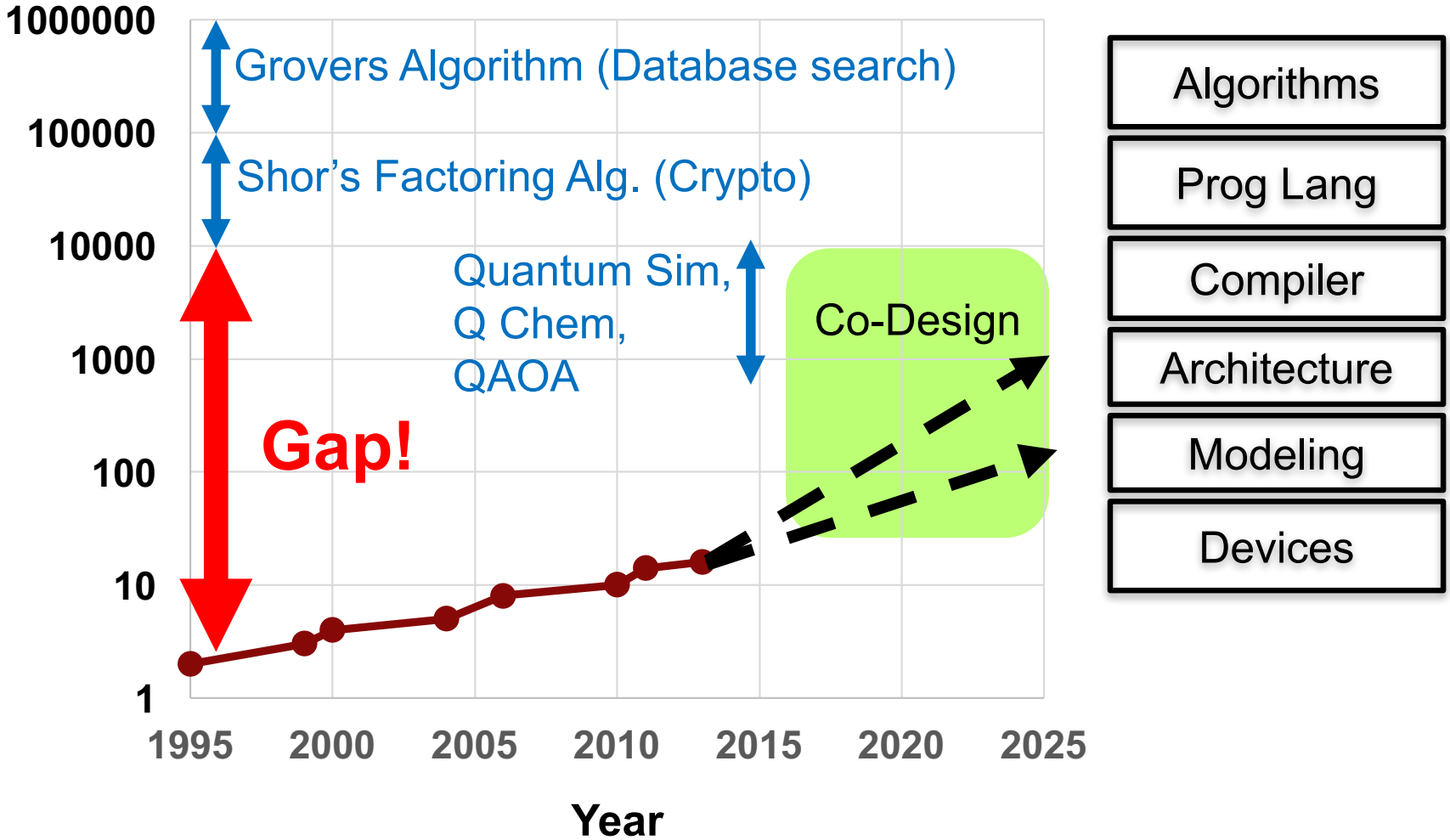
The Algorithms to Machines Gap



The Algorithms to Machines Gap

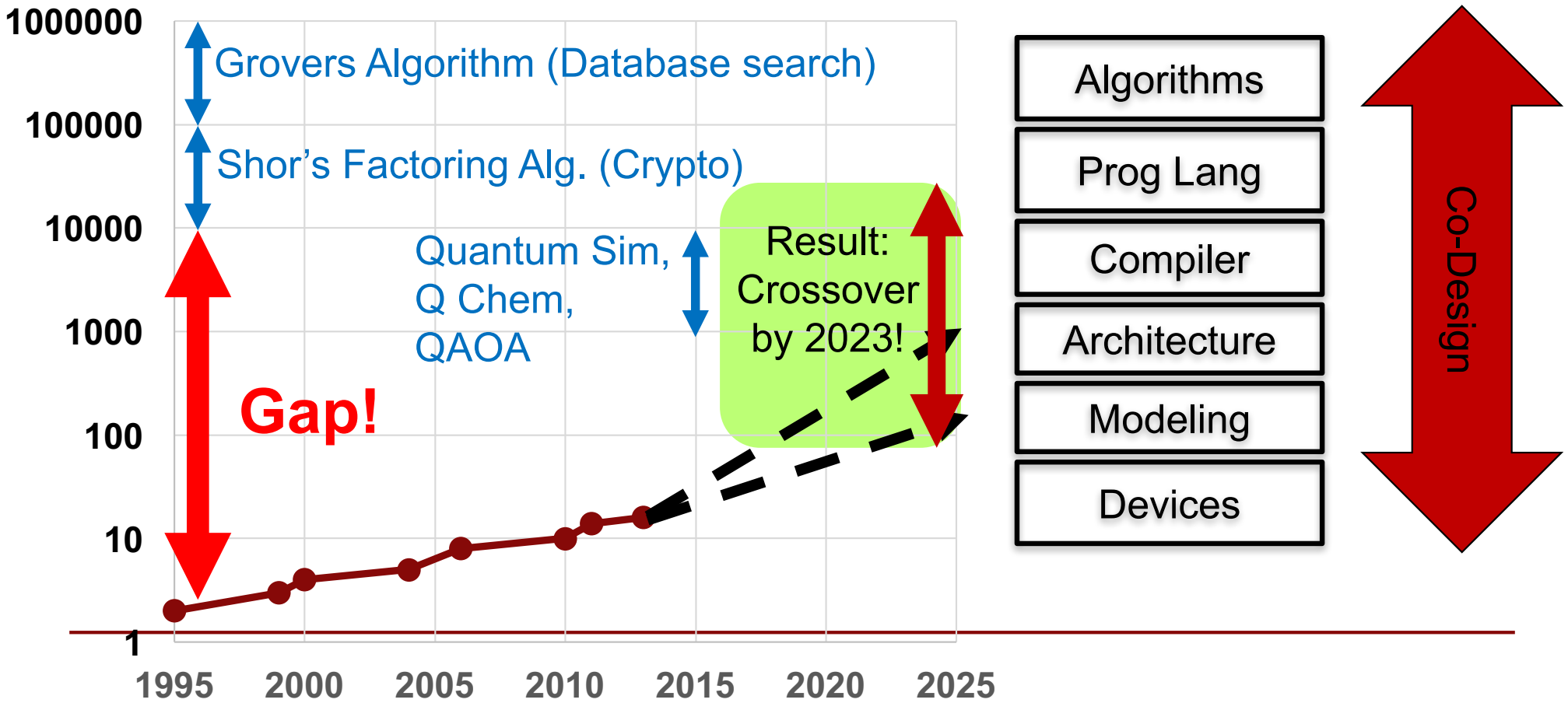


Closing the Gap: Software-Enabled Vertical Integration and Co-Design

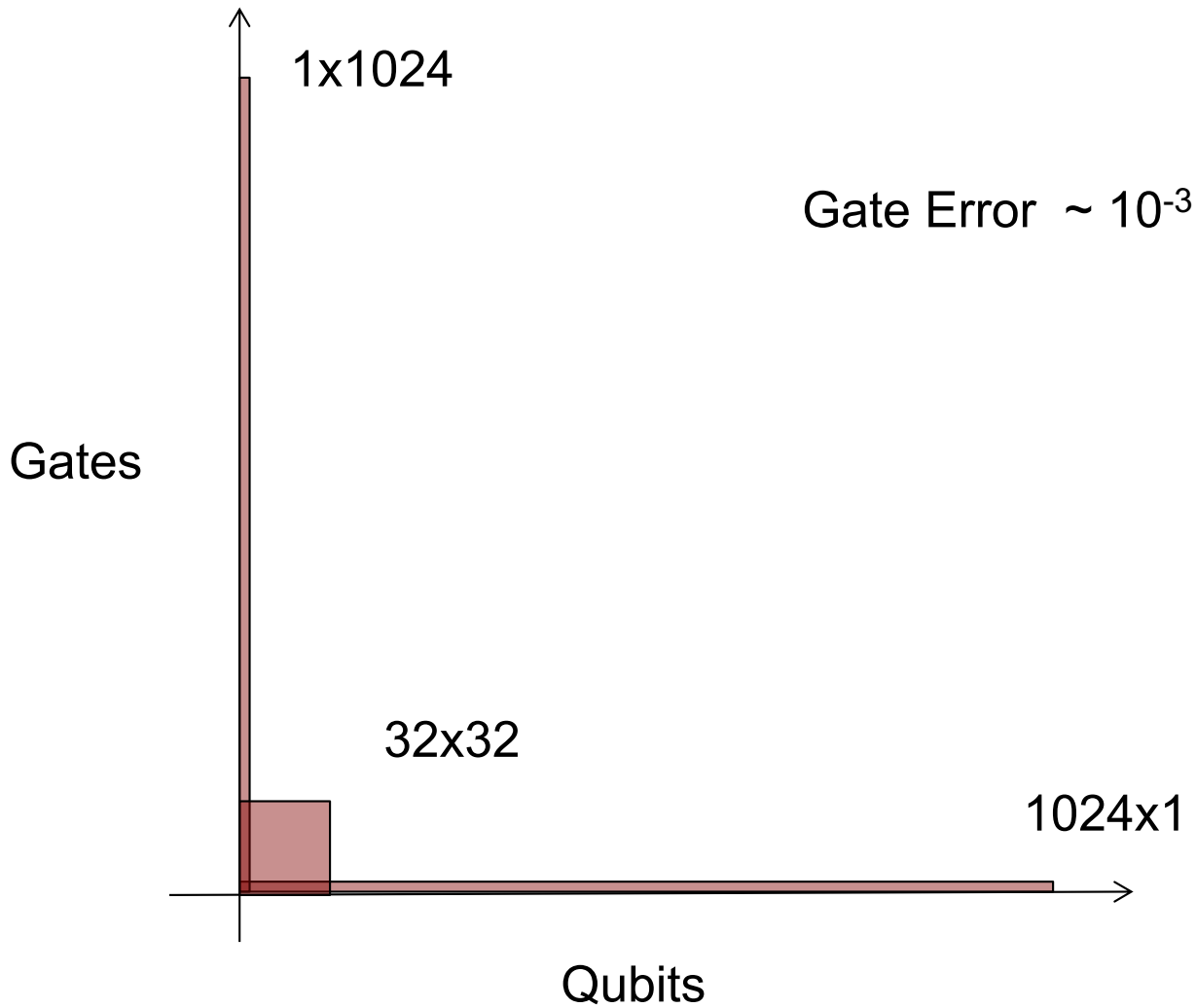


Goal

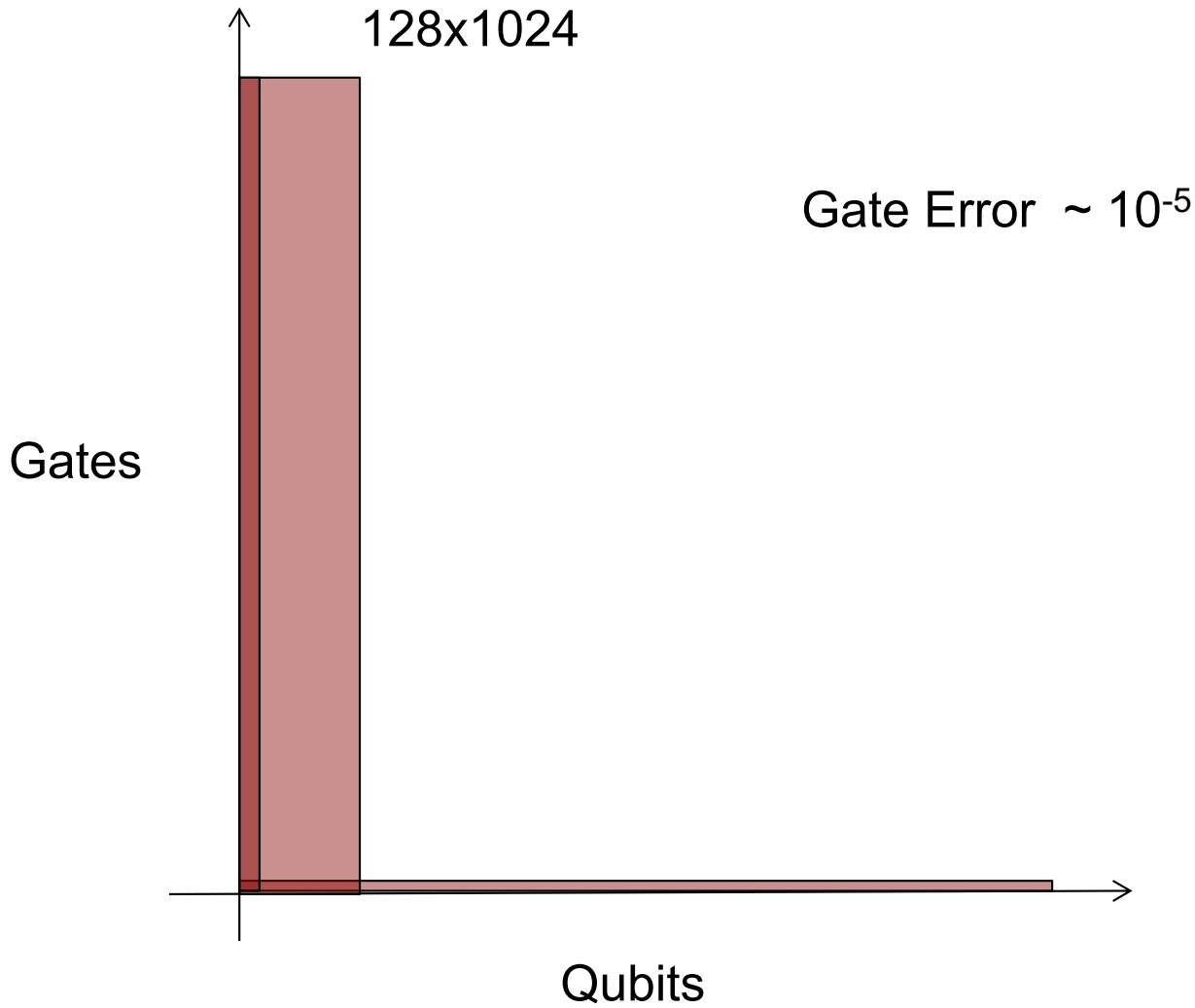
Develop co-designed algorithms, SW, and HW to close the gap between algorithms and devices by 100-1000X, accelerating QC by 10-20 years.



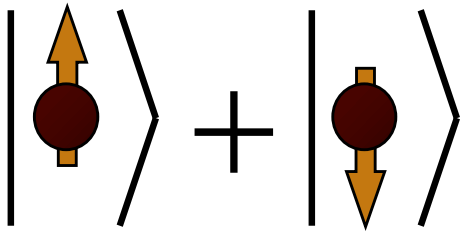
Space-Time Product Limits



Space-Time Product Limits



Quantum Bits (qubit)



- 1 qubit probabilistically represents 2 states

$$|a\rangle = C_0|0\rangle + C_1|1\rangle$$

- Every additional qubit doubles # states

$$|ab\rangle = C_{00}|00\rangle + C_{01}|01\rangle + C_{10}|10\rangle + C_{11}|11\rangle$$

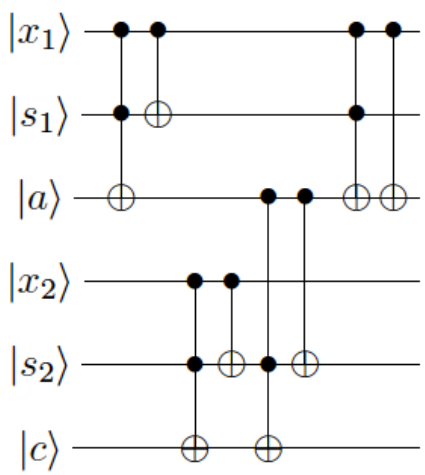
- *Quantum parallelism* on an exponential number of states
- But measurement collapses qubits to single classical values

“Good” Quantum Applications

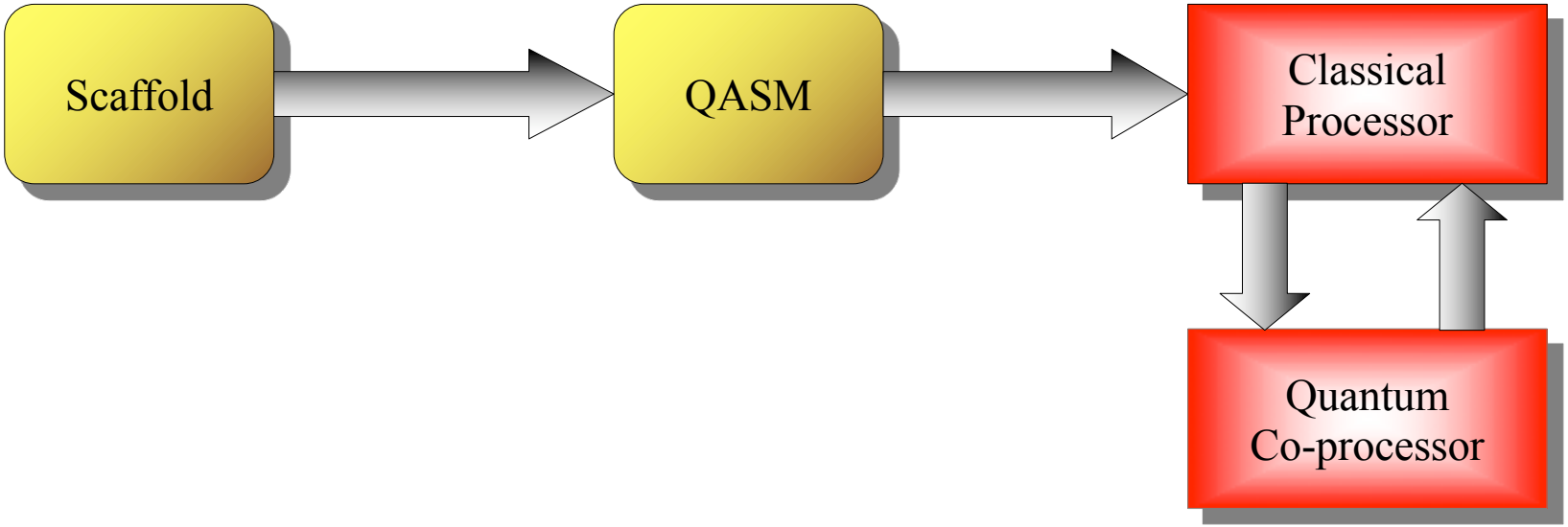
- Compact problem representation
  - Functions, small molecules, small graphs
- High complexity computation
- Compact solution
- Easily-verifiable solution
- Co-processing with classical supercomputers
- Can exploit a small number of quantum kernels

Quantum Compiler Optimizations

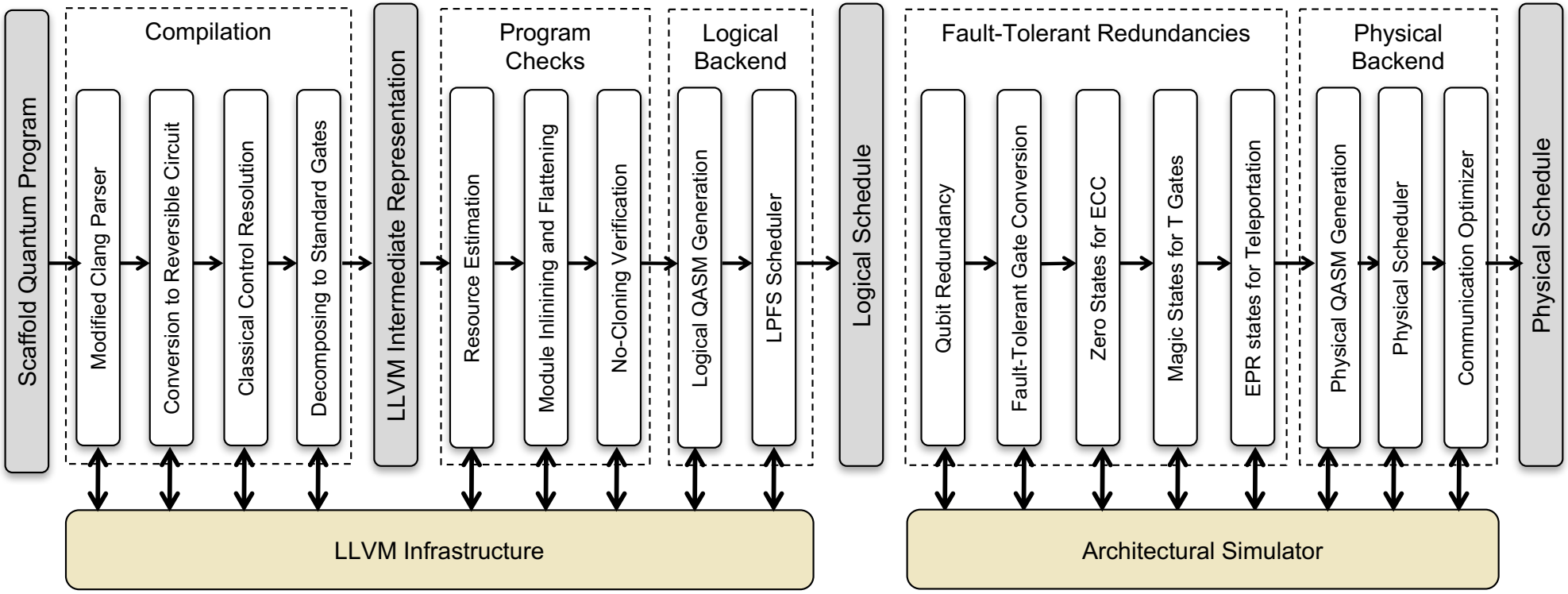
- Similar to circuit synthesis for classical ASICs
- Program inputs often known at compile time
- Manage errors and precision
- Scarce resources
  - Every qubit and gate is important



Execution Model



Tool Flow



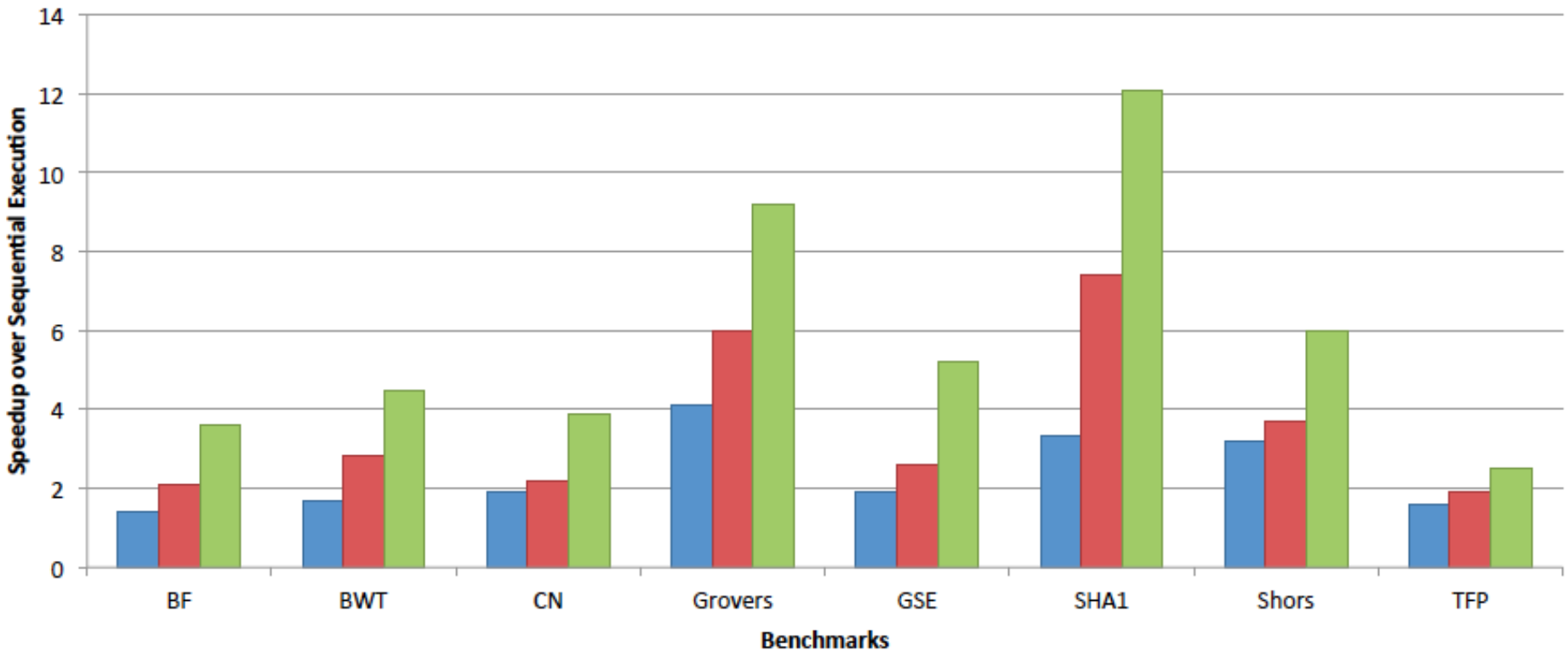
Scaffold tools, 41K lines of code, open source
epiqc.cs.uchicago.edu

https://github.com/epiqc/ScaffCC

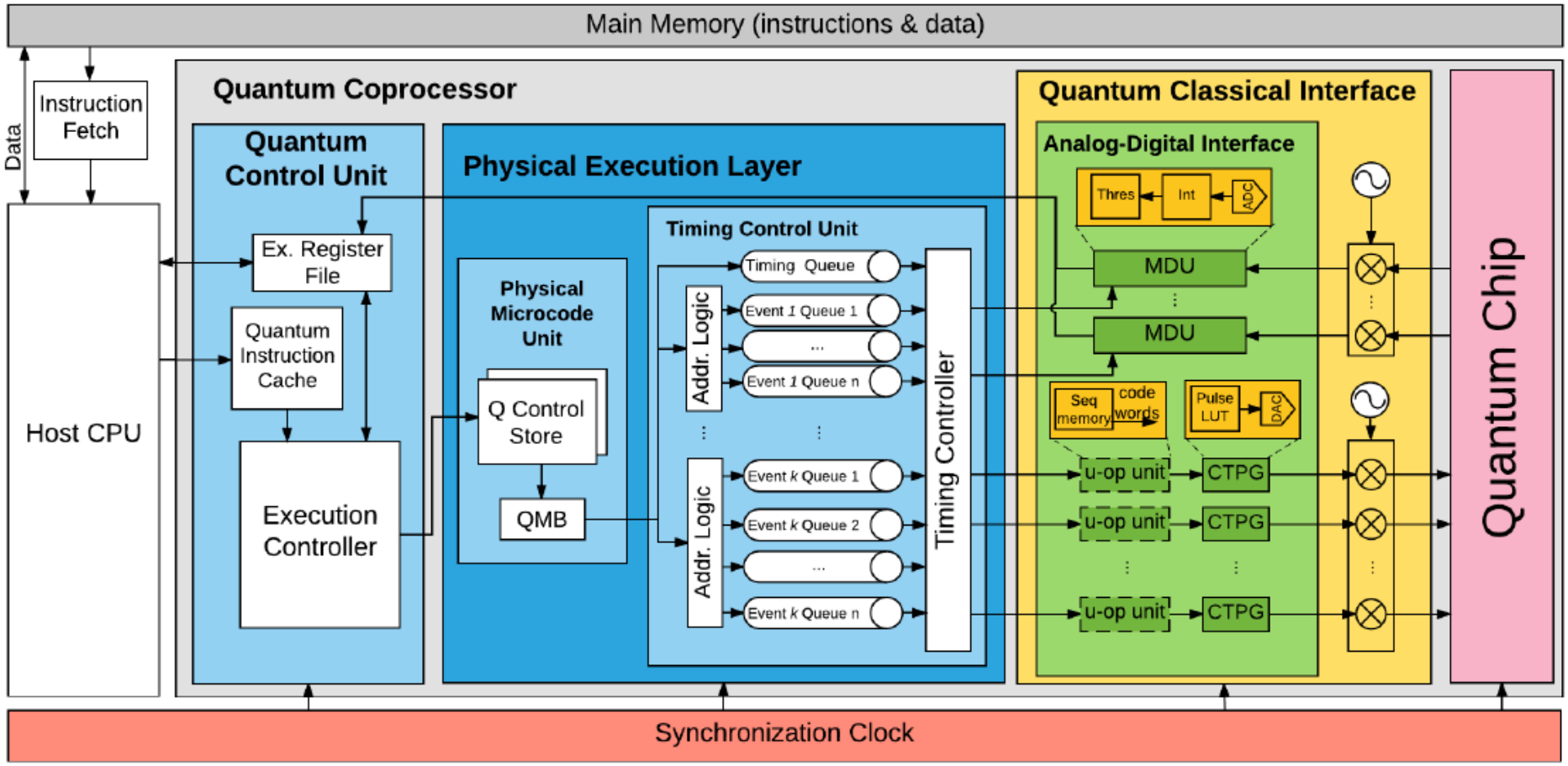
Increasing Parallelism

- Compiler Optimizations:
  - Loop unrolling, constant propagation, inlining, function cloning, DAG scheduling

[Heckey+ ASPLOS 2015]

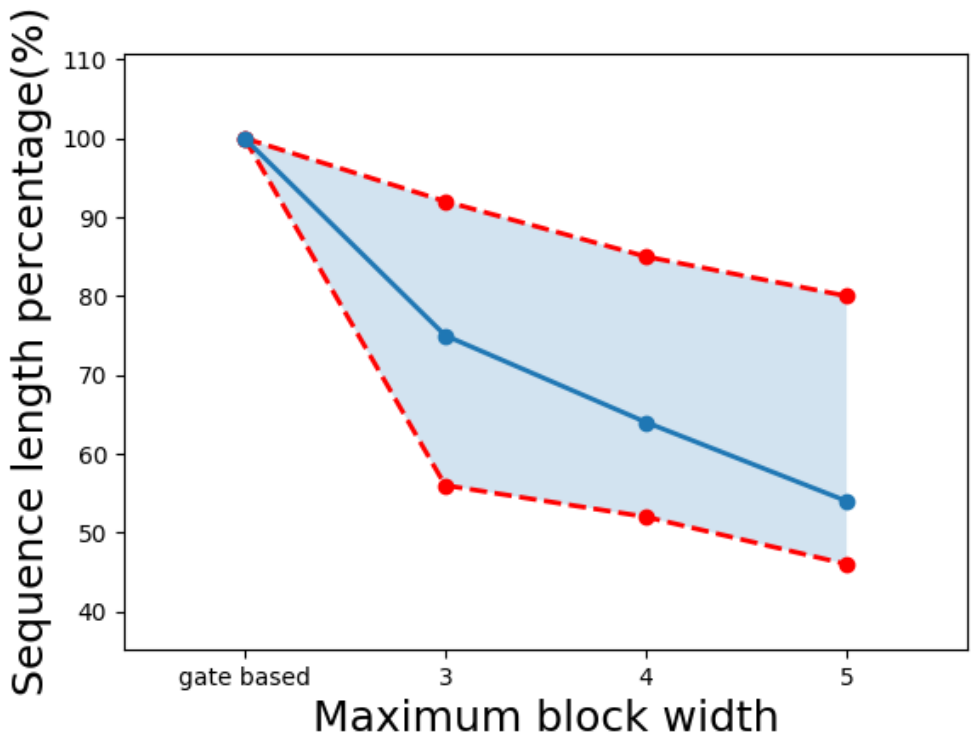


Microarchitecture



[Fu+ Micro 2017 Best Paper]

Breaking ISA Abstraction

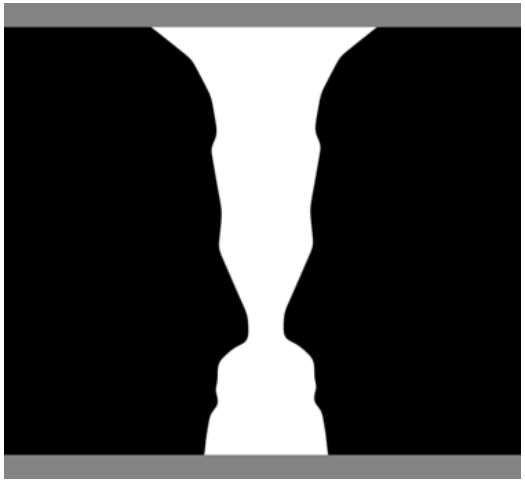


- Multi-Qubit Operators for QAOA
  - Direct translation from compiler to control pulses

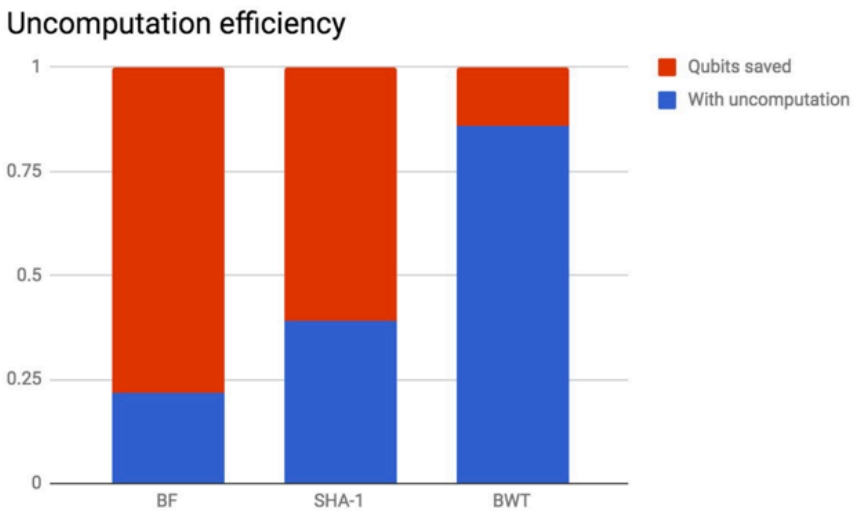
[Joint work with David Schuster]

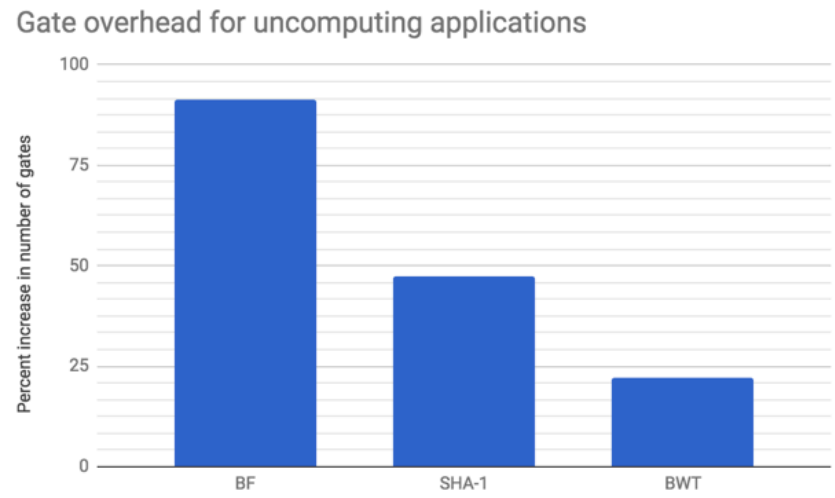
Quantum Memory Management

- Quantum computations use a lot of ancilla
  - Reversible arithmetic
    - NAND -> 3-in-3-out
  - Indirect measurements
  - Error correction
- Ancilla can be reused
  - After measurement
  - After uncomputation
- Tradeoff between qubit and gate usage

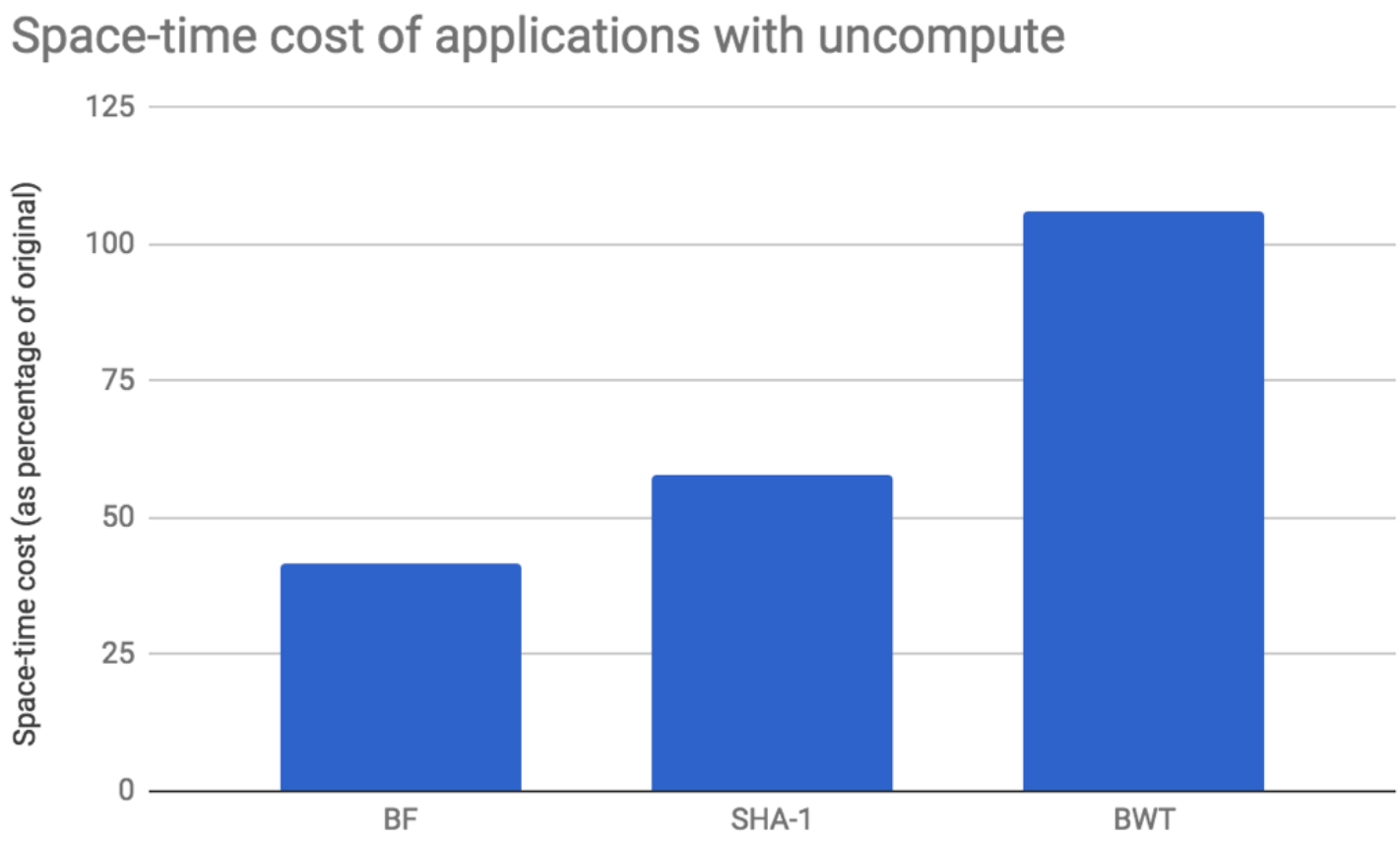


Qubit Savings vs Gate Overhead



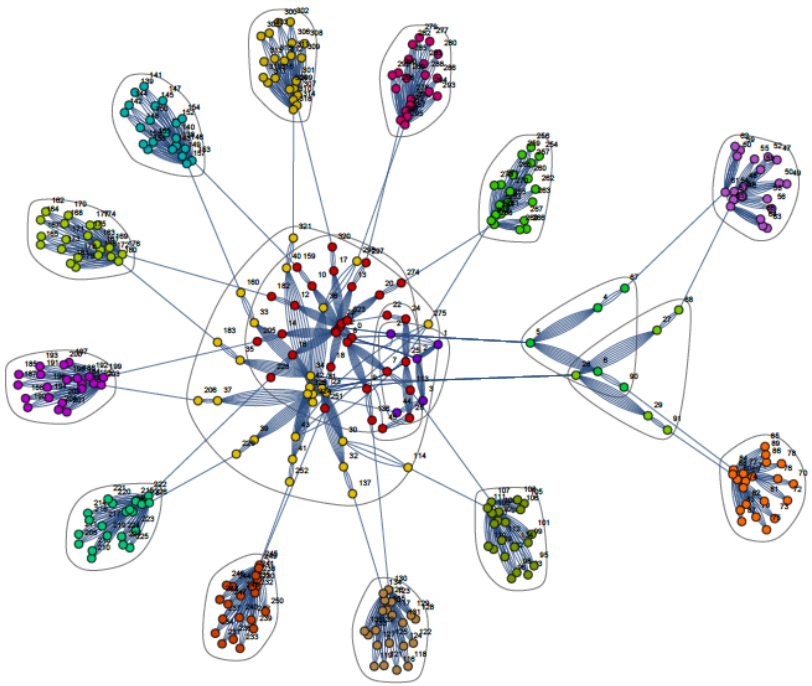


Space-Time Product



How do we map data?

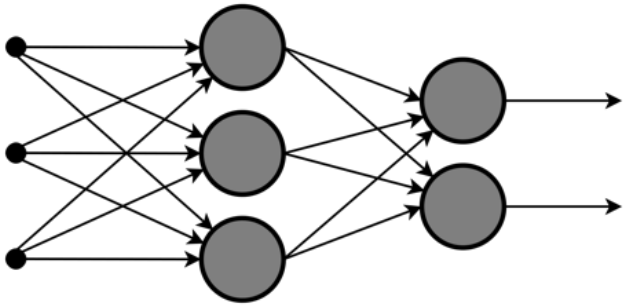
- Static spectral and graph partitioners
- Map for clustering
  - Probably necessary to get to 1000 qubits
- Map for irregular physical constraints
  - Qubit couplings, hardware defects
- Granularity of mappings
- Interaction with qubit reuse



Spectral communities for 2-level Bravyi-Haah magic-state factory

How do we control noise?

- Repeated executions [Temme+ PRL17]
  - Powers of 2 gate times (measure nth-order noise)
  - Random gate errors (simulate noise)
- Machine learning [Mavadia+ NatureComm17]
  - Need to minimize measurements
- Machine learning + control algorithms?

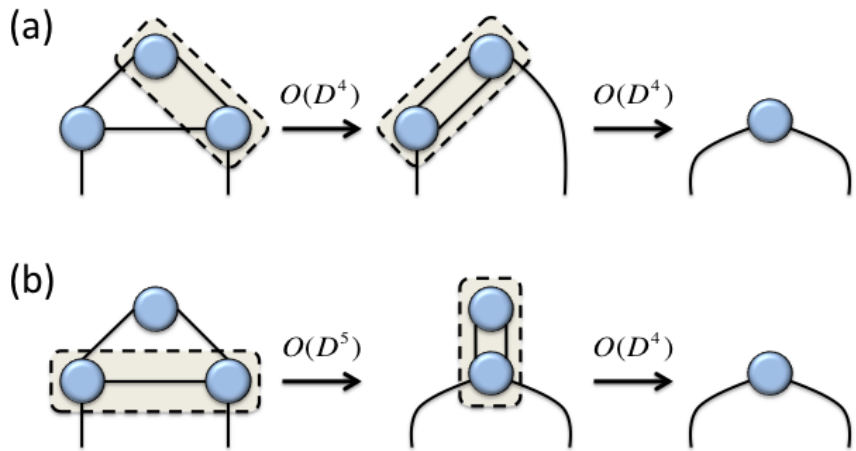


How do I know if my QC program is correct?

- Check implementation against a formal specification
- Check general quantum properties
  - No-cloning, entanglement, uncomputation
- Checks based on programmer assertions (quantum simulation)
- Heuristic bug-finding systems

  [Altadmri SIGCSE15]
- Can we check useful properties in polynomial time for programs with quantum supremacy?

Tensor Contraction in Quantum Simulation



- Without tensors: 45 qubits [Haner+ SC17]
- Find ordering of contractions to most rapidly decrease graph complexity
  - Results in optimized ordering of quantum simulation
  - 49 qubit shallow circuits [Pednault arXiv:1710.05867]

Partial Simulation

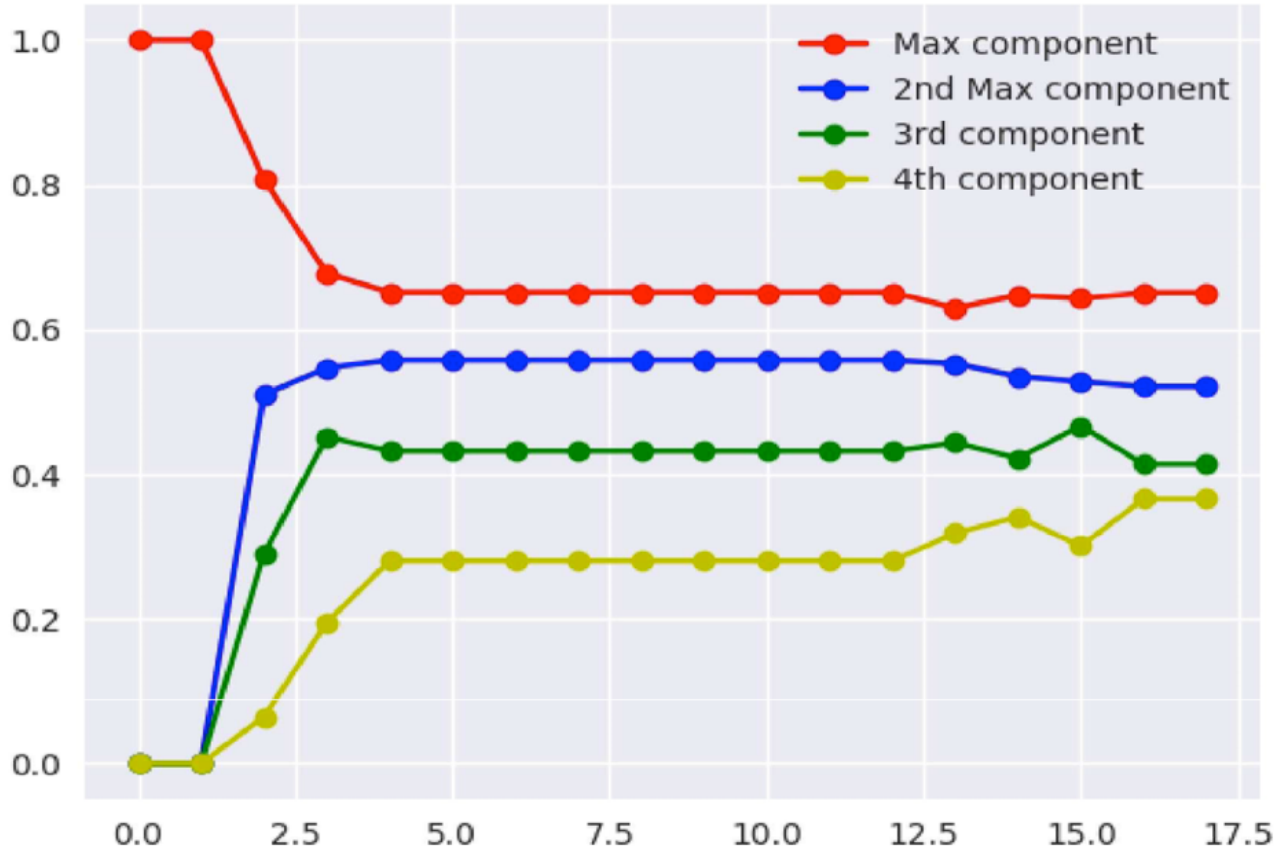
- Polynomial simulation of Clifford + small number of T gates [Bravyi PRL16]
- Polynomial simulation of a specific instance given known input and output?
  - Allows simulation of test vectors

How can tools help the programmer?

- Visualize properties (eg entanglement)
- Visualize qubit usage and memory leaks
- Library building blocks and templates
- Program synthesis techniques

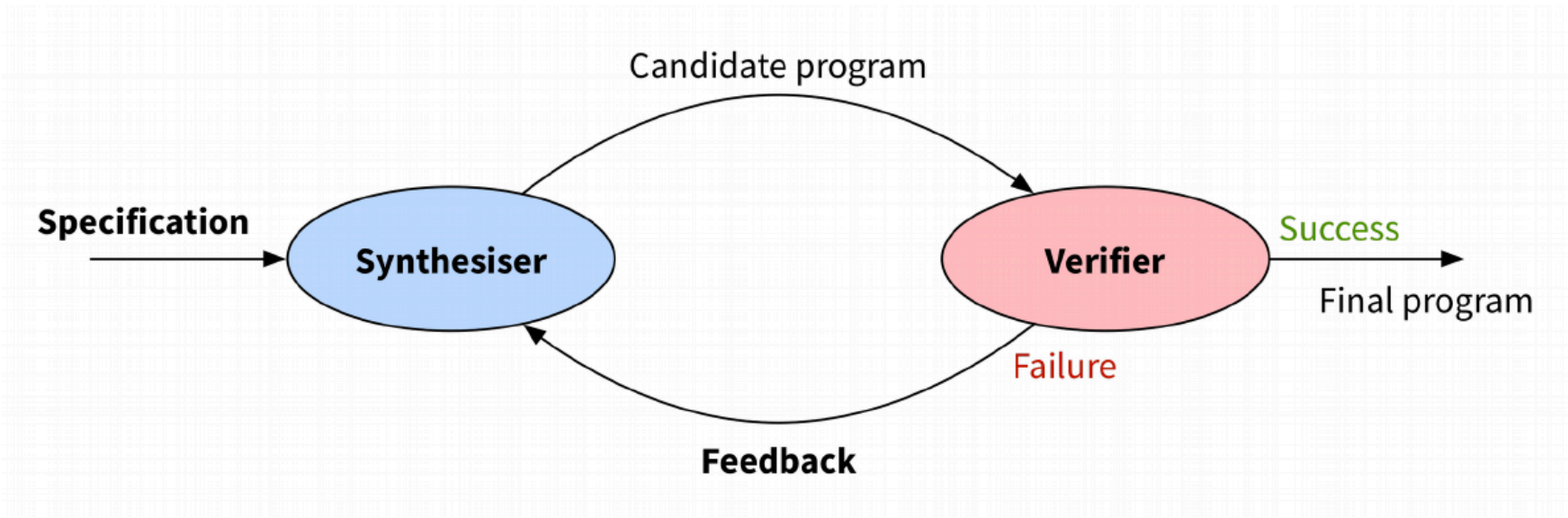
IBM Q Experience

HHL (Linear Systems) Entanglement



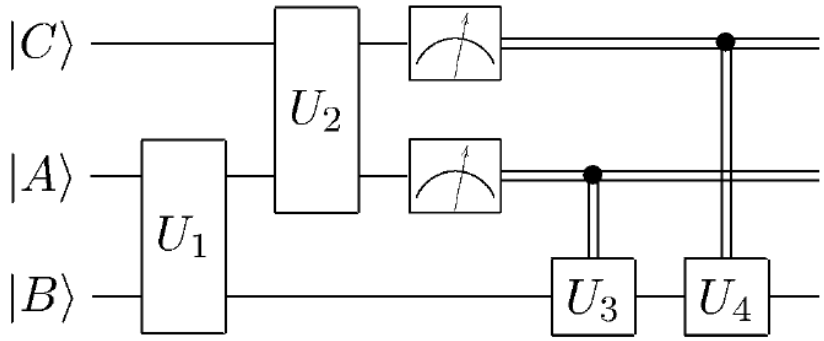
Program Synthesis by Sketching

Counter-example guided synthesis (CEGS)



[Solar-Lezama ASPLOS 06]

Quantum Teleportation



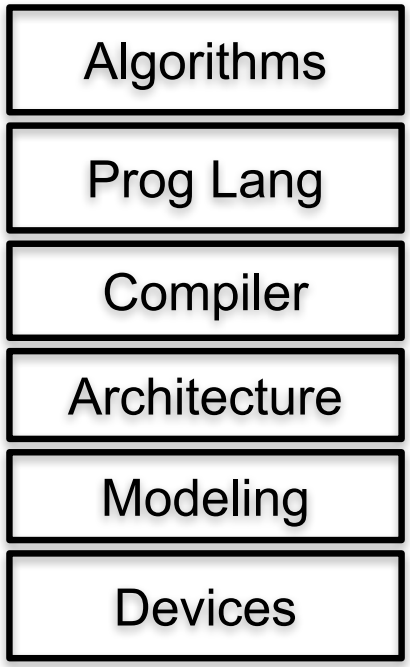
- A family of protocols!

$$U_1 = U_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 0.8 & -\frac{1}{\sqrt{2}} & 0.6 & \frac{1}{\sqrt{2}} \\ -0.6 & \frac{1}{\sqrt{2}} & 0.8 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0.8 & -\frac{1}{\sqrt{2}} & 0.6 \\ \frac{1}{\sqrt{2}} & 0.6 & \frac{1}{\sqrt{2}} & -0.8 \end{pmatrix}$$

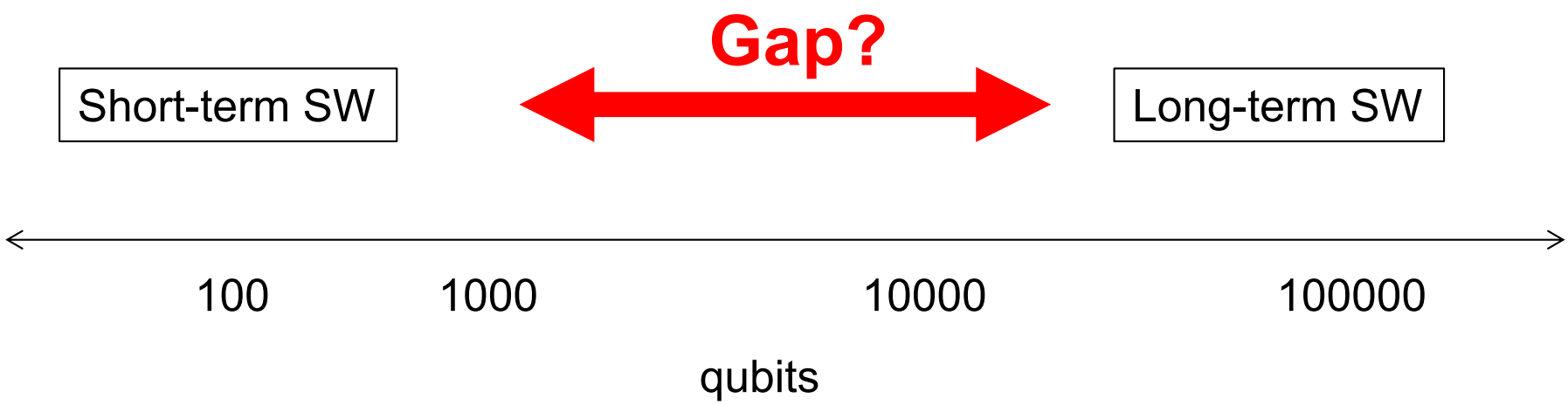
$$U_3 = 4\begin{pmatrix} 0.07 & 0.49 \\ -0.49 & 0.07 \end{pmatrix}^{-1}, U_4 = 4\begin{pmatrix} 0.01 & 0.565 \\ -0.565 & 0.01 \end{pmatrix}^{-1}, U_5 = 4\begin{pmatrix} 0.424 & 0.07 \\ 0.07 & -0.424 \end{pmatrix}^{-1}, U_6 = 4\begin{pmatrix} 0.49 & -0.07 \\ 0.07 & 0.49 \end{pmatrix}^{-1}$$

What are the right abstractions?

- Specification Languages
  - Coq, Hamiltonians
- Programming Languages
  - Scaffold, Quipper, Q#, Quil …
- Instruction-Set Architectures
  - OpenQASM
- Physical Control
  - OpenPulse

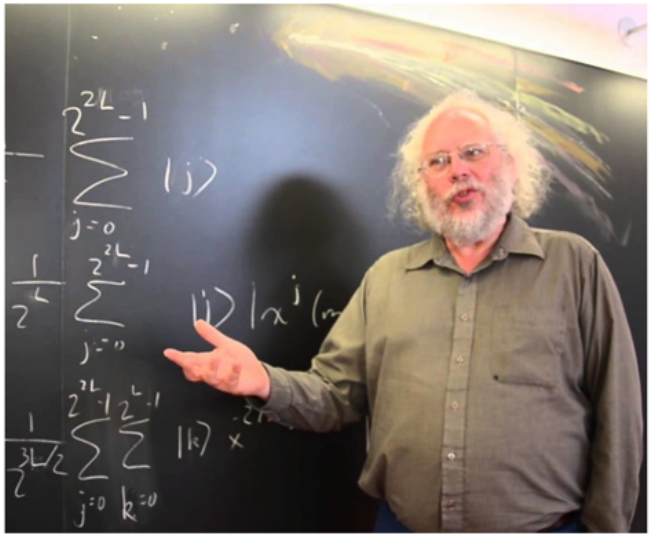


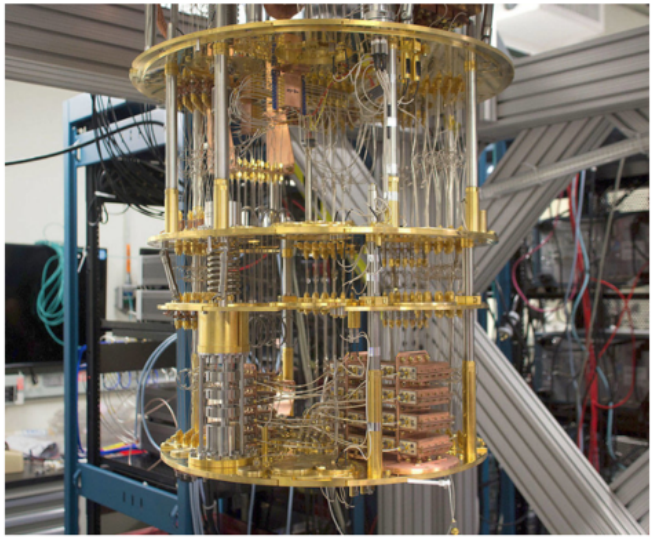
Specialization vs Abstraction



ISCA 2018 Tutorial

GRAND CHALLENGES AND RESEARCH TOOLS FOR QUANTUM COMPUTING





epiqc.cs.uchicago.edu

Summary

- QC is at a historic time
- Software and architecture can generate key insights and accelerate progress
- With the right models and abstractions, classical techniques can have significant impact