Quantum computing: a brief introduction and some research topics

Frank (Peng) Fu

Computer Science and Engineering Department, UofSC

Basics of quantum computing: qubits

- Bit. $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$.
- Qubit. $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$, where $\alpha, \beta \in \mathbb{C}, |\alpha|^2 + |\beta|^2 = 1$.
- Multi-qubits are represented by a tensor product.

$$|\phi_1\rangle \otimes |\phi_2\rangle = (\alpha_1|0\rangle + \beta_1|1\rangle) \otimes (\alpha_2|0\rangle + \beta_2|1\rangle)$$
$$= \alpha_1\alpha_2|00\rangle + \alpha_1\beta_2|01\rangle + \beta_1\alpha_2|10\rangle + \beta_1\beta_2|11\rangle$$

Or,

$$\begin{bmatrix} \alpha_1 \\ \beta_1 \end{bmatrix} \otimes \begin{bmatrix} \alpha_2 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} \alpha_1\alpha_2 \\ \alpha_1\beta_2 \\ \beta_1\alpha_2 \\ \beta_1\beta_2 \end{bmatrix}.$$

Basics of quantum computing: quantum gates

One way to update qubits is via *unitary operations*.

- A complex matrix is unitary if $UU^\dagger = U^\dagger U = I$.
- Linearity: $U(\alpha|0\rangle + \beta|1\rangle) = \alpha U|0\rangle + \beta U|1\rangle$.

Quantum gates

- Hadamard gate.

$$H|0\rangle = |+\rangle = 1/\sqrt{2}(|0\rangle + |1\rangle)$$
$$H|1\rangle = |-\rangle = 1/\sqrt{2}(|0\rangle - |1\rangle)$$

- Phase gate.

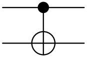
$$S|0\rangle = |0\rangle$$
$$S|1\rangle = i|1\rangle$$

- T gate.

$$T|0\rangle = |0\rangle$$
$$T|1\rangle = \omega|1\rangle, \text{where } \omega^2 = i$$

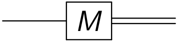
- CNOT gate.

$$\mathrm{CNOT}|00\rangle = |00\rangle, \mathrm{CNOT}|01\rangle = |01\rangle$$
$$\mathrm{CNOT}|10\rangle = |11\rangle, \mathrm{CNOT}|11\rangle = |10\rangle$$



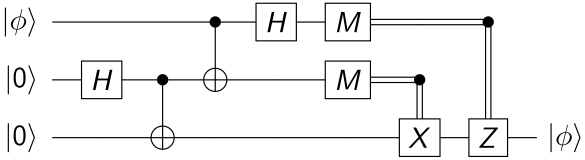
Measurement

Measurement is used to obtain the result of a computation.



- $M(\alpha|0\rangle + \beta|1\rangle) = |0\rangle$ with probability $|\alpha|^2$.
- $M(\alpha|0\rangle + \beta|1\rangle) = |1\rangle$ with probability $|\beta|^2$.

Quantum circuit diagram



Note that $Z = S^2$ and $X = HZH$ is the not-gate.

Fun facts

- The gate set $\{H, S, CNOT\}$ is called *Clifford gate set*. Quantum circuits consists of only Clifford gates can be efficiently simulated using classical computer.
- Clifford gates + T gate is *universal*, i.e., any unitary operations can be approximated by Clifford + T up to epsilon.
- T gates are known to be hard to implement, they dominate the cost of running a quantum circuit.
- Programming with only Clifford gates + T is an art, in practice people also work with a few extra gates, e.g., rotation gates, Toffoli gate.

Research topic 1: T-count optimization

How to reduce the number of T gates or other resource intensive gates (e.g. rotation, toffoli) in a given circuit?

- Existing methods: gate cancellation (e.g., $T^2 = S$), path polynomials (for rotations and CNOT circuits).
- Recent work: *Automated optimization of large quantum circuits with continuous parameters*, Nam et al. 2018.

Research topic 2: Circuit identity verification

Given two circuits, how do we know they are equal?

- Related to Topic 1, how do we know the optimized circuit is correct?
- Existing methods: matrix/algebraic formulation, path-sum formalism (Amy 2018), circuit equational theory (Clément et al. 2023).

Research topic 3: Hamiltonian simulation

- Physicist says: natural evolves according to Schrodinger's equation.
$$v' = -iHv$$
- The solution to the above differential equation is the following.
$$v = e^{-iH} v_0,$$
where H is called *Hamiltonian*, it is a hermitian matrix, i.e., $H = H^\dagger$.
- Hamiltonian simulation problem: Given a Hamiltonian H, find a quantum circuit that implements e^{-iH}.
- In practice H may be sparse but can be exponentially large, it is hard to implement e^{-iH} classically.
- Existing methods: Product formulas (Suzuki 1991), Quantum Signal Processing (Low and Chuang 2017).

Research topic 4: Quantum resource analysis

How much resource does it take to run a useful quantum algorithm?

- Resource: number of qubits, number of T-gates, total running time (circuit depth), hardware resources etc.
- Existing tools: BenchQ (Zapata AI), QUALTRAN (Google Quantum AI), Azure Quantum Resource Estimator (Microsoft Quantum), pyLIQTR (MIT Lincoln Lab), Symbolic QRE Engine (PsiQuantum).

Research topic 5: Quantum programming languages

How do we design and build infrastructure to provide some guarantees about our programs before sending it to quantum computer?

- Qiskit (IBM).
- Cirq (Google).
- Quipper.
- Proto-Quipper.

Research topic 6: Programming quantum computer

Our department has access to IBM's 127 qubits quantum computers. What kind of programs can we run on them to illustrate the usefulness of quantum computer?
Some potential candidates:

- Quantum Arithmetics.
- Quantum Fourier Transform.
- Other possibilities?

Research topic 7: Quantum Computing and AI?

Quantum machine learning?