Logical Abstractions for Noisy Variational Quantum Algorithm Simulation

Yipeng Huang (Rutgers), Steven Holtzen (UCLA), Todd Millstein (UCLA), Guy Van den Broeck (UCLA), Margaret Martonosi (Princeton)

March 10, 2021
What this talk is about:

Using classical probabilistic inference techniques as an abstraction for quantum computing.

- A new way to represent noisy quantum circuits as probabilistic graphical models.
- A new way to encode quantum circuits as conjunctive normal forms and arithmetic circuits.
- A new way to manipulate quantum circuits using logical equation satisfiability solvers.
- Improved simulation and sampling performance for important near-term quantum algorithms.

The Hadamard gate:

\[
\begin{align*}
q_{0m0} &= |0\rangle \text{ AND } q_{0m1} = |0\rangle \rightarrow +\frac{1}{\sqrt{2}} \\
q_{0m0} &= |0\rangle \text{ AND } q_{0m1} = |1\rangle \rightarrow +\frac{1}{\sqrt{2}} \\
q_{0m0} &= |1\rangle \text{ AND } q_{0m1} = |0\rangle \rightarrow +\frac{1}{\sqrt{2}} \\
q_{0m0} &= |1\rangle \text{ AND } q_{0m1} = |1\rangle \rightarrow -\frac{1}{\sqrt{2}} \\
q_{0m2} &= |0\rangle \\
q_{1m3} &= |0\rangle \\
q_{0m2} &= |1\rangle \\
q_{1m3} &= |1\rangle
\end{align*}
\]
Where we are going:

What are quantum variational algorithms?
• Why are they different and important?

What is quantum circuit simulation?
• Why are the conventional techniques insufficient?

How do we represent quantum circuits as logic formulas?
• Why does it help with variational algorithm simulation, and by how much?
Where we are going:

**What are quantum variational algorithms?**
- Why are they different and important?

**What is quantum circuit simulation?**
- Why are the conventional techniques insufficient?

**How do we represent quantum circuits as logic formulas?**
- Why does it help with variational algorithm simulation, and by how much?
NISQ systems target variational algorithms.

Near-term Intermediate Scale Quantum (NISQ) systems have ~100 qubits with at best 0.1% error rate.

With that capacity and reliability, error correction, along with famous algorithms such as Grover’s search and Shor’s factoring are infeasible.

The soonest candidates for useful quantum computation involve quantum-classical variational algorithms.
Hybrid quantum-classical variational algos

Use quantum & classical computation

Quantum evaluates an objective function
Classical optimizes for better parameters

It's like using a classical computer to train a quantum neural network.

Image source: Peruzzo et al., 2013
Specific examples of variational algorithms

Variational quantum eigensolver (VQE)
Simulate quantum mechanics.

Quantum approximate optimization algorithm (QAOA)
Approximate solutions to constraint satisfaction problems (CSPs).
Where we are going:

What are quantum variational algorithms?
• Why are they different and important?

**What is quantum circuit simulation?**
• Why are the conventional techniques insufficient?

How do we represent quantum circuits as logic formulas?
• Why does it help with variational algorithm simulation, and by how much?
The unique challenge of simulating noisy variational algorithms

1. Needs to simulate noise, and quantum circuits are wide but shallow
2. Require repeated simulation with different parameters
3. Only need samples, not full wavefunctions.

Image source: Peruzzo et al., 2013
Rock, paper, scissors: Existing simulation techniques are not suited for variational algorithms

- Schrödinger simulation
- QuEST, IBM, Google; parallel matrix vector multiplication
Schrödinger quantum circuit simulation

\[
\text{CNOT(H } \otimes I|00\rangle) = \text{CNOT(H|0\rangle} \otimes I|0\rangle) = \text{CNOT}
\begin{bmatrix}
\frac{1}{\sqrt{2}} & 1 \\
1 & \frac{1}{\sqrt{2}}
\end{bmatrix}
\begin{bmatrix}
1 \\
0
\end{bmatrix}
= 
\begin{bmatrix}
1 & 1 \\
1 & 1
\end{bmatrix}
\begin{bmatrix}
1/\sqrt{2} \\
0 \\
0 \\
1/\sqrt{2}
\end{bmatrix}
= \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle
\]
Rock, paper, scissors: Existing simulation techniques are not suited for variational algorithms

<table>
<thead>
<tr>
<th>Schrödinger simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>QuEST, qSim, ...; parallel matrix vector multiplication</td>
</tr>
</tbody>
</table>

1. Does it excel at simulating wide but shallow circuits? 
   - ✗

2. Does it extract structure for repeated simulation with different parameters? 
   - ✗

3. Does it efficiently sample from the final wavefunction? 
   - ✓
Rock, paper, scissors: Existing simulation techniques are not suited for variational algorithms

<table>
<thead>
<tr>
<th></th>
<th>Schrödinger simulation</th>
<th>Feynman simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>QuEST, qSim, …; parallel matrix vector multiplication</td>
<td>✗</td>
<td>qTorch; graphical model tensor network contraction</td>
</tr>
</tbody>
</table>

1. Does it excel at simulating wide but shallow circuits?  
   - Schrödinger simulation: ✗  
   - Feynman simulation: ✗  

2. Does it extract structure for repeated simulation with different parameters?  
   - Schrödinger simulation: ✗  
   - Feynman simulation: ✗  

3. Does it efficiently sample from the final wavefunction?  
   - Schrödinger simulation: ✓  
   - Feynman simulation: ✓
The image shows a Feynman quantum circuit simulation. The circuit includes qubits labeled as q0m0, q0m1, q0m2, q1m1, and q1m2, with transitions indicated by arrows and quantum gates. The circuit diagram illustrates the evolution of qubits under different quantum operations.

The table provides the transformation of qubits from initial states to final states under the operation of the quantum circuit. The table is structured as follows:

| q0m1 = |0⟩ | q0m1 = |1⟩ | q1m1 = |0⟩ | q1m1 = |1⟩ |
|---|---|---|---|---|
| q1m1 = |0⟩ | 1 | 0 | 0 | 0 |
| q1m1 = |1⟩ | 0 | 1 | 0 | 0 |
| q0m2 = |0⟩ | 0 | 0 | 0 | 1 |
| q0m2 = |1⟩ | 0 | 0 | 1 | 0 |

The left side of the circuit shows the initial state of qubits, and the right side shows the state after applying the quantum gate and evaluation function. The quantum gate transformation matrix is also provided, which is used to calculate the final state of the qubits.

The circuit demonstrates the principles of quantum computing, such as superposition and entanglement, and how they are used to simulate quantum systems.
Rock, paper, scissors: Existing simulation techniques are not suited for variational algorithms

<table>
<thead>
<tr>
<th>Schrödinger simulation</th>
<th>Feynman simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>QuEST, IBM, Google; parallel matrix vector multiplication</td>
<td>qTorch; graphical model tensor network contraction</td>
</tr>
</tbody>
</table>

1. Does it excel at simulating wide but shallow circuits?
   - Schrödinger: ✗
   - Feynman: ✓

2. Does it extract structure for repeated simulation with different parameters?
   - Schrödinger: ✗
   - Feynman: ?

3. Does it efficiently sample from the final wavefunction?
   - Schrödinger: ✓
   - Feynman: ✗
Where we are going:

What are quantum variational algorithms?
• Why are they different and important?

What is quantum circuit simulation?
• Why are the conventional techniques insufficient?

How do we represent quantum circuits as logic formulas?
• Why does it help with variational algorithm simulation, and by how much?
Our toolchain: Bayesian network knowledge compilation for noisy quantum circuit simulation and sampling

1. Noisy quantum circuits to Bayesian network
2. Bayesian networks to conjunctive normal form (CNF)
3. CNF to arithmetic circuit (AC)
4. Exact inference on AC for quantum circuit simulation
5. Gibbs sampling on AC to sample from final wavefunction
Our toolchain: Bayesian network knowledge compilation for noisy quantum circuit simulation and sampling

1. Noisy quantum circuits to Bayesian network
2. Bayesian networks to conjunctive normal form (CNF)
3. CNF to arithmetic circuit (AC)
4. Exact inference on AC for quantum circuit simulation
5. Gibbs sampling on AC to sample from final wavefunction

1. Needs to simulate noise
2. Repeated simulation with different parameters
3. Only need samples, not full wavefunctions
Our toolchain: Bayesian network knowledge compilation for noisy quantum circuit simulation and sampling

1. Noisy quantum circuits to Bayesian network

2. Bayesian networks to conjunctive normal form (CNF)

3. CNF to arithmetic circuit (AC)

4. Exact inference on AC for quantum circuit simulation

5. Gibbs sampling on AC to sample from final wavefunction
Bayesian networks: AI models that encode probabilistic knowledge in a factorized format

<table>
<thead>
<tr>
<th>RAIN</th>
<th>SPRINKLER</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>0.4</td>
</tr>
<tr>
<td>T</td>
<td>0.01</td>
</tr>
</tbody>
</table>

| RAIN | 0.2 | 0.8 |

<table>
<thead>
<tr>
<th>SPRINKLER</th>
<th>RAIN</th>
<th>0.0</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>0.8</td>
<td>0.2</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>0.9</td>
<td>0.1</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>0.99</td>
<td>0.01</td>
</tr>
</tbody>
</table>

Image source: Wikimedia
Noisy quantum circuits to Bayesian network

| q0m0   | P( q0m1=|0⟩ ) | P( q0m1=|1⟩ ) |
|--------|-------------|-------------|
| |0⟩ | +1/√2 | +1/√2 |
| |1⟩ | +1/√2 | −1/√2 |

| Control q0m2 | Target q1m0 | P( q1m3=|0⟩ ) | P( q1m3=|1⟩ ) |
|--------------|-------------|-------------|-------------|
| |0⟩ | |0⟩ | 1. | 0. |
| |0⟩ | |1⟩ | 0. | 1. |
| |1⟩ | |0⟩ | 0. | 1. |
| |1⟩ | |1⟩ | 1. | 0. |
Our toolchain: Bayesian network knowledge compilation for noisy quantum circuit simulation and sampling

1. Noisy quantum circuits to Bayesian network

2. **Bayesian networks to conjunctive normal form (CNF)**

3. CNF to arithmetic circuit (AC)

4. Exact inference on AC for quantum circuit simulation

5. Gibbs sampling on AC to sample from final wavefunction
Bayesian networks to conjunctive normal form (CNF)

Think about circuit as logic equation

Compile & minimize this logic equation

Variable assignments that satisfy CNF are valid Feynman paths through algorithm
• Model count on variable assignments yields quantum circuit simulation
Bayesian networks to conjunctive normal form (CNF)

Qubits take on binary values:

\[ q_{0m0} = |0> \text{ XOR } q_{0m0} = |1> \]
\[ q_{0m1} = |0> \text{ XOR } q_{0m1} = |1> \]
\[ q_{0m2} = |0> \text{ XOR } q_{0m2} = |1> \]
\[ q_{1m0} = |0> \text{ XOR } q_{1m0} = |1> \]
\[ q_{1m3} = |0> \text{ XOR } q_{1m3} = |1> \]
Bayesian networks to conjunctive normal form (CNF)

The Hadamard gate:

\[
\begin{align*}
q_{0m0} &= |0\rangle \quad \text{AND} \quad q_{0m1} = |0\rangle \quad \rightarrow \quad +1/\sqrt{2} \\
q_{0m0} &= |0\rangle \quad \text{AND} \quad q_{0m1} = |1\rangle \quad \rightarrow \quad +1/\sqrt{2} \\
q_{0m0} &= |1\rangle \quad \text{AND} \quad q_{0m1} = |0\rangle \quad \rightarrow \quad +1/\sqrt{2} \\
q_{0m0} &= |1\rangle \quad \text{AND} \quad q_{0m1} = |1\rangle \quad \rightarrow \quad -1/\sqrt{2}
\end{align*}
\]

| \( q_{0m0} \) | \( P( q_{0m1}=|0\rangle ) \) | \( P( q_{0m1}=|1\rangle ) \) |
|---|---|---|
| \( |0\rangle \) | \( +1/\sqrt{2} \) | \( +1/\sqrt{2} \) |
| \( |1\rangle \) | \( +1/\sqrt{2} \) | \( -1/\sqrt{2} \) |
Bayesian networks to conjunctive normal form (CNF)

The CNOT gate:

- \( q0m2 = |0\rangle \text{ AND } q1m0 = |0\rangle \rightarrow q1m3 = |0\rangle \)
- \( q0m2 = |0\rangle \text{ AND } q1m0 = |1\rangle \rightarrow q1m3 = |1\rangle \)
- \( q0m2 = |1\rangle \text{ AND } q1m0 = |0\rangle \rightarrow q1m3 = |1\rangle \)
- \( q0m2 = |1\rangle \text{ AND } q1m0 = |1\rangle \rightarrow q1m3 = |0\rangle \)

| Control \( q0m2 \) | Target \( q1m0 \) | \( P( q1m3 = |0\rangle ) \) | \( P( q1m3 = |1\rangle ) \) |
|----------------|----------------|----------------|----------------|
| \( |0\rangle \) | \( |0\rangle \) | 1. | 0. |
| \( |0\rangle \) | \( |1\rangle \) | 0. | 1. |
| \( |1\rangle \) | \( |0\rangle \) | 0. | 1. |
| \( |1\rangle \) | \( |1\rangle \) | 1. | 0. |
Bayesian networks to conjunctive normal form (CNF)

Put all the sentences together!

Convert logical implications "→" to logical disjunctions

Conjoin all the disjunctive clauses together to form CNF (i.e., AND all the ORs together)
Our toolchain: Bayesian network knowledge compilation for noisy quantum circuit simulation and sampling

1. Noisy quantum circuits to Bayesian network

2. Bayesian networks to conjunctive normal form (CNF)

3. \textbf{CNF to arithmetic circuit (AC)}

4. Exact inference on AC for quantum circuit simulation

5. Gibbs sampling on AC to sample from final wavefunction
CNF to arithmetic circuit (AC)

Figure 1: Equivalent knowledge compilation representations of a 4-qubit noisy QAOA quantum circuit. In this work we calculate and sample amplitudes from arithmetic circuits (ACs) representing noisy quantum circuits. To the left, direct compilation results in ACs where qubit states ordered in time increases from top to bottom. Above, logical minimization, variable reordering, and eliding internal states reduces the size of the AC. The equivalent reduced representation leads to more efficient simulation and sampling.

Figure 6: Simulation resource requirements vs. quantum circuit size for three quantum algorithms
Our toolchain: Bayesian network knowledge compilation for noisy quantum circuit simulation and sampling

1. Noisy quantum circuits to Bayesian network

2. Bayesian networks to conjunctive normal form (CNF)

3. CNF to arithmetic circuit (AC)

4. **Exact inference on AC for quantum circuit simulation**

5. Gibbs sampling on AC to sample from final wavefunction
Exact inference on AC for quantum circuit simulation

- Quantum simulation becomes tree traversal on AC
Quantum simulation becomes tree traversal on AC

Quantum measurement outcomes are probabilistic evidence
Exact inference on AC for quantum circuit simulation

- Quantum simulation becomes tree traversal on AC
- Quantum measurement outcomes are probabilistic evidence
  - **Amplitude for given outcome comes from root node**
Our toolchain: Bayesian network knowledge compilation for noisy quantum circuit simulation and sampling

1. Noisy quantum circuits to Bayesian network

2. Bayesian networks to conjunctive normal form (CNF)

3. CNF to arithmetic circuit (AC)

4. Exact inference on AC for quantum circuit simulation

5. **Gibbs sampling on AC to sample from final wavefunction**
Gibbs sampling on AC to sample from final wavefunction

\[
\begin{align*}
q_0 m_2 r.v &= 0 \quad |0>, 0, 0.8/\sqrt{2} \\
q_0 m_2 r.v &= 1 \quad |1>, 0.6, -0.6/\sqrt{2} \\
q_1 m_3 &= 0 \quad |0>, 1, -0.6/\sqrt{2} \\
q_1 m_3 &= 1 \quad |1>, 1, -0.6/\sqrt{2} \\
\end{align*}
\]

Table 10: Downward pass for finding derivatives for Gibbs sampling MCMC

<table>
<thead>
<tr>
<th>Present sample</th>
<th>( q_0 m_2 r.v )</th>
<th>( q_0 m_2 )</th>
<th>( q_1 m_3 )</th>
<th>amplitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gibbs sample noise</td>
<td>0</td>
<td></td>
<td></td>
<td>0.8</td>
</tr>
<tr>
<td>Gibbs sample qubits</td>
<td>1</td>
<td>0</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Gibbs sample qubits</td>
<td>1</td>
<td></td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
Schrödinger quantum circuit simulation

\[ \text{CNOT}(H \otimes I|00\rangle) = \text{CNOT}(H|0\rangle \otimes I|0\rangle) = \text{CNOT} \begin{bmatrix} \frac{1}{\sqrt{2}} & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} \\ 0 \end{bmatrix} = \begin{bmatrix} 1/\sqrt{2} \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle \]
Feynman quantum circuit simulation

\[ \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \]

| q0m0 = |0⟩ | q0m0 = |1⟩ | q0m1 = |0⟩ | q0m1 = |1⟩ |
|---|---|---|---|
| q1m0 = |0⟩ | 1  | 0  | 0  | 0  |
| q1m0 = |1⟩ | 0  | 1  | 0  | 0  |
| q1m2 = |0⟩ | 0  | 0  | 0  | 1  |
| q1m2 = |1⟩ | 0  | 0  | 1  | 0  |

| q0m0 = |0⟩ | q0m0 = |1⟩ | q0m1 = |0⟩ | q0m1 = |1⟩ |
|---|---|---|---|
| q1m0 = |0⟩ | 1/√2 | 0  | 1/√2 | 0  |
| q1m0 = |1⟩ | 0  | 1/√2 | 0  | 1/√2 |
| q1m2 = |0⟩ | 0  | 1/√2 | 0  | -1/√2 |
| q1m2 = |1⟩ | 1/√2 | 0  | -1/√2 | 0  |
Where we are going.

What are quantum variational algorithms?
• Why are they different and important?

What is quantum circuit simulation?
• Why are the conventional techniques insufficient?

How do we represent quantum circuits as logic formulas?
• Why does it help with variational algorithm simulation, and by how much?
Where we are going.

What are quantum variational algorithms?
• Why are they different and important?

What is quantum circuit simulation?
• Why are the conventional techniques insufficient?

How do we represent quantum circuits as logic formulas?
• *Why does it help with variational algorithm simulation, and by how much?*
The unique challenge of simulating noisy variational algorithms

1. Needs to simulate noise, and quantum circuits are wide but shallow

2. Require repeated simulation with different parameters

3. Only need samples, not full wavefunctions.

Image source: Peruzzo et al., 2013
Our toolchain: Bayesian network knowledge compilation for noisy quantum circuit simulation and sampling

1. Noisy quantum circuits to Bayesian network

2. Bayesian networks to conjunctive normal form (CNF)

3. CNF to arithmetic circuit (AC)

4. Exact inference on AC for quantum circuit simulation

5. Gibbs sampling on AC to sample from final wavefunction

1. Needs to simulate noise

2. Repeated simulation with different parameters

3. Only need samples, not full wavefunctions
Result 1: It works!

With minimal modification, knowledge compilation exact inference can be repurposed for quantum simulation

• Can accurately simulate Pauli gates, CNOT, CZ, phase kickback, Toffoli, CHSH protocol, Deutsch-Jozsa, Bernstein-Vazirani, hidden shift, quantum Fourier transform, Shor’s, Grover’s…

• Passes Google Cirq’s suite of test harness for quantum simulators
Result 2: Ideal circuit simulation

![QAOA simulation time vs. qubits (iterations=1)](chart)

- qsim sampling with 1 thread
- qsim sampling with 16 threads
- qtorch sampling with 1 thread
- qtorch sampling with 16 threads
- Knowledge compilation sampling

Qubits, representing Max-Cut problem vertices
Result 2: Noisy circuit simulation

Noisy QAOA simulation time vs. qubits (iterations=1)

- density matrix sampling
- knowledge compilation sampling
What this talk was about:

Using classical probabilistic inference techniques as an abstraction for quantum computing.
- A new way to represent noisy quantum circuits as probabilistic graphical models.
- A new way to encode quantum circuits as conjunctive normal forms and arithmetic circuits.
- A new way to manipulate quantum circuits using logical equation satisfiability solvers.
- Improved simulation and sampling performance for important near-term quantum algorithms.

The Hadamard gate:

\[
\begin{align*}
q_0m_0 &= |0\rangle \quad \text{AND} \quad q_0m_1 = |0\rangle \quad \rightarrow +\frac{1}{\sqrt{2}} \\
q_0m_0 &= |0\rangle \quad \text{AND} \quad q_0m_1 = |1\rangle \quad \rightarrow +\frac{1}{\sqrt{2}} \\
q_0m_0 &= |1\rangle \quad \text{AND} \quad q_0m_1 = |0\rangle \quad \rightarrow +\frac{1}{\sqrt{2}} \\
q_0m_0 &= |1\rangle \quad \text{AND} \quad q_0m_1 = |1\rangle \quad \rightarrow -\frac{1}{\sqrt{2}}
\end{align*}
\]
Where we have gone:

What are quantum variational algorithms?
• Why are they different and important?

What is quantum circuit simulation?
• Why are the conventional techniques insufficient?

How do we represent quantum circuits as logic formulas?
• Why does it help with variational algorithm simulation, and by how much?
Broader research agenda: new representations for quantum computing

**Schrödinger**: state vectors and density matrices

**Heisenberg**: stabilizer formalism

**Feynman**: tensor-network path sums

**Binary decision diagrams** (new?)

**Logical satisfiability equations** (this work; new?)
<table>
<thead>
<tr>
<th>Ongoing work: accelerating quantum research via graphical model inference techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1</strong> Improving the correctness of quantum algorithms.</td>
</tr>
<tr>
<td>• Genetic algorithms on graphical model topologies for improving variational ansatzes.</td>
</tr>
<tr>
<td>• Graphical model training to understand variational algorithm progress.</td>
</tr>
<tr>
<td><strong>2</strong> Benchmarking the performance of quantum programs on unreliable quantum prototypes.</td>
</tr>
<tr>
<td>• Bayesian network sensitivity analysis for debugging misbehaving program implementations.</td>
</tr>
<tr>
<td>• Bayesian network knowledge compilation to enable more efficient repeated simulation.</td>
</tr>
<tr>
<td><strong>3</strong> Probing the limits of classical computing to better understand the potential of quantum computing.</td>
</tr>
<tr>
<td>• Classical hardware accelerators for graphical models in support of quantum simulation.</td>
</tr>
</tbody>
</table>
Thank you to my collaborators

Margaret Martonosi; Princeton

Steven Holtzen, Todd Millstein, Guy Van den Broeck; UCLA

Members of the EPiQC team

• This work is funded in part by EPiQC, an NSF Expedition in Computing, under grant 1730082.