Ongoing Projects on Quantum Circuits and Algorithms

Igor L. Markov and John P. Hayes
Advanced Computer Architecture Laboratory
EECS Department
University of Michigan,
Ann Arbor, MI 48109

Ongoing Projects

- Simulation of quantum circuits
  - BDD-based QuIDDPro simulator
  - Simulating Grover’s algorithm
- Synthesis of two-qubit circuits
  - Bounds for gate counts in two-qubit circuits
- Quantum algs that improve memory usage
  - Quantum counters

Quantum Circuit Simulation Using QuIDDs

- Motivation
  - Need for a better way to simulate quantum circuits
- Quantum Information Decision Diagram (QuIDD)
  - Novel data representation that uses Binary Decision Diagrams (BDD) widely used in computer-aided circuit design
  - Captures some exponentially-sized matrices and vectors in a form that grows polynomially with the number of qubits
  - Multiplies matrices and vectors in compressed form
- QuIDDPro Simulator
  - Our QuIDD-based simulator implemented in C++
  - Experiments with Grover’s algorithm demonstrate fast execution and low memory utilization

QuIDD Data Representation

```
00 01 10 11
00 1/2 1/2 1/2 1/2
01 1/2 -1/2 1/2 -1/2
10 1/2 1/2 -1/2 -1/2
11 1/2 -1/2 -1/2 1/2
```
**QuIDD Data Representation**

\[
\begin{bmatrix}
00 & 1/2 & 1/2 & 1/2 \\
01 & 1/2 & -1/2 & 1/2 \\
10 & 1/2 & 1/2 & 1/2 \\
11 & 1/2 & -1/2 & -1/2
\end{bmatrix}
\]

**QuIDDPro Simulation of Grover’s Algorithm**

- Search for items in an unstructured database of \(N\) items
- Contains \(n = \log N\) qubits and has runtime

**QuIDDPro Simulation Results (Grover’s search algorithm)**

<table>
<thead>
<tr>
<th>Oracle</th>
<th>Runtime (s)</th>
<th></th>
<th>PMU (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(n)</td>
<td>Oct</td>
<td>MAT</td>
<td>D++</td>
</tr>
<tr>
<td>10</td>
<td>3.06c-2</td>
<td>2.06c-2</td>
<td>1.06c-2</td>
</tr>
<tr>
<td>11</td>
<td>2.94c-2</td>
<td>4.02</td>
<td>0.72</td>
</tr>
<tr>
<td>12</td>
<td>2.82c-2</td>
<td>3.32c-2</td>
<td>2.22</td>
</tr>
<tr>
<td>13</td>
<td>2.70c-3</td>
<td>2.02e-3</td>
<td>6.92</td>
</tr>
<tr>
<td>14</td>
<td>2.68e-3</td>
<td>3.90e-3</td>
<td>2.39</td>
</tr>
<tr>
<td>15</td>
<td>2.68e-4</td>
<td>5.90e-4</td>
<td>10.1</td>
</tr>
<tr>
<td>16</td>
<td>TIME-OUT</td>
<td>TIME-OUT</td>
<td>2.13c-2</td>
</tr>
<tr>
<td>17</td>
<td>TIME-OUT</td>
<td>TIME-OUT</td>
<td>8.46c-2</td>
</tr>
<tr>
<td>18</td>
<td>TIME-OUT</td>
<td>TIME-OUT</td>
<td>1.92c-2</td>
</tr>
<tr>
<td>19</td>
<td>TIME-OUT</td>
<td>TIME-OUT</td>
<td>3.74c-3</td>
</tr>
<tr>
<td>20</td>
<td>TIME-OUT</td>
<td>TIME-OUT</td>
<td>1.74c-3</td>
</tr>
</tbody>
</table>

**QuIDDPro Simulation of Grover’s Algorithm**

Same results for any oracle that distinguishes a unique element
QuIDDPro Simulation of Grover’s Algorithm

- Database search with a black-box predicate $p(x) = 1$
  - Classical evaluation of $p(x)$ on one input (queries)
  - Quantum (parallel) evaluation of $p(x)$ facilitates an implementation with fewer queries
- We also assume that $p(x)$ is given as a BDD/QuIDD
  - BDDs are used to represent functions in practical CAD
  - However, a BDD is not really a black-box
  - BDD operations evaluate $p(x)$ on multiple inputs at once (no quantum computation is involved)
- Grover on QuIDDs: same query complexity as in the quantum case
  - In practice this simulation is very fast and needs little memory

Quantum Circuit Synthesis

- Synthesis of classical circuits
  - Given a truth table, it is easy to find a circuit
  - Gate-count minimization is trickier, but doable by hand for circuits with several inputs
- Synthesis of $n$-input quantum circuits
  - Given a $2^n \times 2^n$ matrix, can find a circuit (known algorithm)
  - Gate-count minimization doable by hand only for one input
  - For two inputs, optimal constructions are less than one year old, involve taking square roots of $4 \times 4$ matrices…

Two-qubit Computation with Minimum Resources

1. Some elementary gates have 2 inputs; our work allows to compare gate libraries
2. Most physical implementations of q. computers are currently restricted to 2 qubits
3. Circuits for quantum communication often have 2-3 inputs
4. Given a quantum circuit with $>2$ inputs, we can look for 2-input subcircuits and re-optimize those (peephole optimization)
**Universal Elementary Gates [Barenco et.al. ’95]**

- Elementary one-qubit gates:
  \[
  K_x(\theta) = \begin{pmatrix}
  \cos \theta / 2 & \sin \theta / 2 \\
  -\sin \theta / 2 & \cos \theta / 2
  \end{pmatrix}, \quad 0 \leq \theta < 2\pi
  \]
  \[
  K_y(\alpha) = \begin{pmatrix}
  e^{-i\alpha / 2} & 0 \\
  0 & e^{i\alpha / 2}
  \end{pmatrix}, \quad 0 \leq \alpha < 2\pi
  \]

- Elementary two-qubit gates: CNOT, conditioned on any line

- Barenco et al.: CNOT, \(K_x(\theta)\) and \(K_y(\alpha)\) are universal

**Technology-Independent Synthesis**

- **Input:** Unitary \(4 \times 4\)-matrix \(M\)
  - Generic quantum computation on 2 qubits
- **Output:** circuit in terms of elem. gates that implements \(M\) up to a phase
- **Minimize:** circuit cost
  - E.g., gate count or \(\sum\) (gate costs)
- **Solutions exist iff the gate library is universal**

**Small Quantum Circuits**

- What are the worst-case shortest quantum circuits up to phase?
- One-qubit computation: 3 gates required, suffice
- Technique: matrix decompositions

\[
U = \begin{pmatrix}
  e^{i\theta} & 0 & 0 & 0 \\
  0 & e^{-i\theta / 2} & 0 & 0 \\
  0 & 0 & e^{i\theta / 2} & 0 \\
  0 & 0 & 0 & e^{-i\theta / 2}
\end{pmatrix}
\]

Gate 1 Gate 2 Gate 3

**Example 5.3** Let \(J^f\) be the two-qubit Quantum Fourier Transform (QFT) [6]. It is given by the matrix

\[
J^f = \frac{1}{2} \begin{pmatrix}
  1 & 1 & 1 & 1 \\
  1 & -1 & -1 & -1 \\
  1 & -1 & 1 & -1 \\
  1 & 1 & -1 & -1
\end{pmatrix}
\]

This operator is at the heart of Shor’s factoring algorithm, and is one of the few operators whose implementation on a quantum computer is exponentially faster than any known classical counterpart [6]. Choosing the canonical decomposition appropriately and applying Theorem 5.1, one obtains the following circuit, in which \(T = K_x(\pi / 4)\).

---

Advanced Computer Architecture Laboratory

---

Advanced Computer Architecture Laboratory
Previous Work

- Proof of universality is constructive
  - [Barenco et. al. '95] in Phys. Rev. A
  - Can be interpreted as a synthesis algorithm
  - However, no attempt to minimize #gates
- Can be viewed as matrix factorization
  - \( M = QR \) with unitary \( Q \) & upper-triangular \( R \)
    - \( M \) unitary \( \Rightarrow R \) diagonal
- We count gates, and the answer is

Our Results

- New synthesis procedures for 2 qubits
  - Can implement any operator in 18 gates or less, at most 3 of them are CNOTs
  - Lower bounds: sometimes 18 gates and 3 CNOTs are required
  - For a specific operator, we can tell when 0, 1, 2 or 3 CNOTs are required

Our Work (2)

- Lower bounds
  - There exist two-qubit computations (most of them) that require at least 17 elementary gates
    - At least 15 non-const gates
    - At least 2 CNOTs
  - Bounds are not constructive and not tight, except for “15 input-dependent gates”
- We never use “temporary storage” qubits but that could lead to smaller gate counts

The Entangler and Disentangler

- “Computational basis”
  - \( |00\rangle, |01\rangle, |10\rangle \) and \( |11\rangle \)
- The “entangler” computation maps \( |00\rangle \) to \( \frac{|00\rangle + |11\rangle}{\sqrt{2}} \), etc.
- The “disentangler” is \( E^*E \)
- Key lemma
  - If \( U = A \), then \( EUE^* \) has only real entries
  - An efficient way to recognize tensor products
Circuits For $E$ and $E^*$

- A specific circuit for the entangler $E$
  
  ![Circuit](image)

- $S=\text{diag}(1,i)$ counts as one elementary gate
- The Hadamard gate $H$ counts as two
- $E^*$ is implemented by reversing the diagram
  - Change $S$ to $S^*=\text{diag}(1,-i)$

Our (Key) Synthesis Procedure

- The “canonical decomposition” for 2-qubit computations:
  - $\forall U \exists K_1, K_2$ and $\Delta$ such that $U=K_1 \Delta K_2$
  - $E \Delta E^*$ is diagonal (5 gates)
  - $K_1, K_2$ have only real entries
- The terms $K_1, K_2$ and $\Delta$ can be found explicitly
  - Numerical analysis: polar and spectral decompositions
- Reduce $K_1$ and $K_2$ to tensor products using entanglers
  - $EUE^*=E(A \otimes B)E^*$
  - $E(C \otimes D)E^*$
- $A, B, C$ and $D$ are one-qubit computations: 3 gates each
- Note that $E$ and $E^*$ are the same for any input

Details (1)

- After the initial “divide-and-conquer”
  many gate cancellations can be made
- This brings down max #gates to 28
  - Only 15 of them depend on input, which matches an a priori lower bound
- Further reductions based on the analysis of $E(A \otimes B)E^*$ and $E(C \otimes D)E^*$
  - Max no. of gates reduced to
  - However, 19 gates depend on the input

Details (2)

- For additional details, see
  - Physical Review A 68(1), July 2003, 012318 quant-ph/0211002
Validation of Our Synthesis Algorithm

- Implementation in C++
  - We plan to put it up on the Web as an ASP
- Can capture structure
  - Several examples in quant-ph/0211002
  - Optimal results for any $A \otimes B$ circuit
    (QR decomposition $\rightarrow$ typically 61 gates)
  - For 2-qubit Fourier transform;
a circuit with minimal # of CNOT gates

Summary

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Decomposition</th>
<th># Elem. Gates</th>
<th># CNOTs</th>
<th># Var. 1-qubit Gates</th>
</tr>
</thead>
<tbody>
<tr>
<td>QR</td>
<td>61</td>
<td>18</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td>Our #1</td>
<td>$a \otimes b$</td>
<td>25</td>
<td>4</td>
<td>19</td>
</tr>
<tr>
<td>Our #2</td>
<td>$A \otimes B$</td>
<td>28</td>
<td>8</td>
<td>15 (sharp)</td>
</tr>
<tr>
<td>Our lower bounds</td>
<td>17</td>
<td>2</td>
<td>15</td>
<td></td>
</tr>
</tbody>
</table>

- First generic synthesis algorithm to capture circuit structure, e.g., $A \otimes B$.
- Recent work (1)
  - Lower and upper bounds of 61 gates (almost done)
  - Solved the synthesis of $n$-qubit diagonal computations quant-ph/0303039 (asymptotically optimal circuits)

Recent Work (2)

Table 1. Gate structure upper bounds on gate counts for generic circuits using various gate decompositions (4). To compare the upper bounds we use the idea of a generalized qubit gate. The bounds are given in terms of the number of gates that have to be used to implement the circuit.$^{(4)}$

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Circuit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A \otimes B$</td>
<td>$A \otimes B$</td>
<td>CNOT-gate cancellations</td>
</tr>
<tr>
<td>$A \otimes B$</td>
<td>$A \otimes B$</td>
<td>CNOT-gate cancellations</td>
</tr>
<tr>
<td>$A \otimes B$</td>
<td>$A \otimes B$</td>
<td>CNOT-gate cancellations</td>
</tr>
<tr>
<td>$A \otimes B$</td>
<td>$A \otimes B$</td>
<td>CNOT-gate cancellations</td>
</tr>
<tr>
<td>$A \otimes B$</td>
<td>$A \otimes B$</td>
<td>CNOT-gate cancellations</td>
</tr>
</tbody>
</table>

Table 2. Circuit identities used in our work.

Recent Work