Quantum Algorithms & Circuits for Scientific Computing

Anargyros Papageorgiou
Department of Computer Science
Columbia University

Joint work with: M. Bhaskar, S. Hadfield and I. Petras
Overview

• Why quantum algorithms for scientific computing

• Requirements

• Quantum algorithms & circuits for fundamental functions, e.g., $\sqrt{w}$, $\ln w$ etc.
  
  – Algorithms by combining elementary modules

  – Applications

  – Tests

• Summary
Why quantum circuits for scientific computing

• Scientific computing applications can benefit from fast quantum algorithms
  – e.g., numerical linear algebra problems

• Typically we need to compute fundamental functions such as $\sqrt{x}$, $\ln x$, $\sin x$

• Classical computation: IEEE standard for floating point arithmetic
  – Comprehensive math. libraries

• Quantum computation:
  – No standard for numerical computations
  – No general purpose quantum circuits implementing fundamental functions
  – Details about numerical calculations have been avoided in a way
Requirements – quantum circuit model

• Develop a standard for numerical computations
  – Reversible computation

• Fixed precision representation of numbers

• Quantum algorithms for scientific computing with performance guarantees
  – error & cost
Quantum algorithms & circuits for fundamental functions

• Library of elementary quantum circuit templates implementing
  – Arithmetic expressions
  – Shifts
  – Initial approximations for iterative methods
  – New circuits are added to the library as they are derived

• Elementary quantum circuits are modules with known error and cost characteristics

• New algorithms are implemented by combining modules
Algorithms

Input: $n+1$ qubit register

$$|w\rangle = |w_s\rangle \otimes \underbrace{|w^{(m-1)}\rangle \otimes \cdots \otimes |w^{(0)}\rangle \otimes |w^{(-1)}\rangle \otimes \cdots \otimes |w^{(m-n)}\rangle}_{\text{sign integer part fractional part}}$$

$w_s, w^{(j)} \in \{0,1\}, \quad w = (-1)^{w_s} \sum_{j=m-n}^{m-1} w^{(j)} 2^j$

- Register length may be different for intermediate calculations.
Elementary quantum circuit template examples

1.

- Addition and multiplication imply that the format of $res$ is known given the format of the inputs

- $n$ bit inputs: $res$ can be represented exactly using $2n + 1$ bits (qubits) (plus sign) of which $2(n - m)$ hold the fractional part

- Any desired number $b$ of significant bits after the decimal point in $res$ can be passed on to the next stage
Quantum circuit which on input $w \geq 1$ computes $\hat{x}_0 = 2^{-p}$ for $p$ such that $2^p > w \geq 2^{p-1}$.

Initial approximation for Newton iteration computing $w^{-1}$, $w \geq 1$

Other similar circuits are also included
Applications

We have derived quantum algorithms for:

- $w^{-1}$
- $\sin w$, $\cos w$
- Inverse trigonometric
- $\sqrt{w}$
- $w^{1/2^i}, i = 1, \ldots, k$
- $\ln w$
- $w^f$, $f \in (0,1)$

Earlier work [Cao, P, Petras, Traub, Kais]
Poisson equation
Square root \(-\sqrt{w}, w \geq 1\)

Use Newton iteration

Selection of function whose zero is \(\sqrt{w}\) is important

e.g., \(f(x) = x^2 - w\) is not a good choice

Iterative step \(x_i = x_{i-1} - (x_{i-1}^2 - w)/(2x_{i-1})\) requires division

Need extra circuitry to keep track of location of decimal point in result
We use:

1. One iteration \( x_i \to w^{-1} \)

\[
x_i = g_1(x_{i-1}) := -w x_{i-1}^2 + 2 x_{i-1}, \quad i = 1, \ldots, s_1
\]

2. Second iteration \( y_j \to \frac{1}{\sqrt{x_{s_1}}} \approx \sqrt{w} \)

\[
y_j = g_2(y_{j-1}) := \frac{3 y_{j-1} - x_{s_1} y_{j-1}^3}{2}, \quad j = 1, \ldots, s_2
\]
Quantum circuit for $\sqrt{w}$,

$\hat{y}_0$, $\hat{x}_0$ init. approx.

In each stage calculations are exact.

Results are truncated to $b$ bits (qubits) after the decimal point and passed on to the next stage.

$w = \begin{array}{c} m \text{ bits} \\ \cdot \\ n - m \text{ bits} \end{array}$
Thm.

\[ |\hat{y}_s - \sqrt{w}| \leq \left(\frac{3}{4}\right)^{b-2m} (2 + b + \log_2 b), \quad b \geq \max\{2m, 4\} \]

Cost:

- # iterative steps: \( s := s_1 = s_2 = O(\log_2 b) \)
- # qubits per step: \( O(n + b) \)
- # quantum ops. per step: low degree poly in \( n + b \)
Logarithm - $\ln w, w > 1$

Algorithm:

1. Shift right $w$ to obtain $w_p = 2^{1-p} w \in (1,2)$, with $2^p > w \geq 2^{p-1}$

2. Compute $t_p = w_p^{1/2^\ell}$. Note $t_p = 1 + \delta$, with $\delta \approx 2^{-\ell}$

3. Approximate $\ln t_p \approx \delta - \frac{\delta^2}{2}$

4. $\ln w \approx (p - 1) \ln 2 + 2^\ell \left( \delta - \frac{\delta^2}{2} \right)$
Quantum circuit for $\ln w$

\[
\begin{align*}
    |w\rangle & \rightarrow \text{Right Shift} \quad p - 1 \text{ times} \quad 1 \leq w_p < 2 \quad |w_p\rangle \\
    |0\rangle & \rightarrow |w_p\rangle \\
    t_p = w_p^{1/2^l} & \quad |\hat{t}_p\rangle \\
    y_p = f(\hat{t}_p) & \quad |\hat{y}_p\rangle \\
    z_p = 2^l \hat{y}_p & \quad |z_p\rangle
\end{align*}
\]

\[
y_p = f(t_p) = (t_p - 1) - \frac{(t_p-1)^2}{2} \approx \ln t_p \quad \text{(step 3 of alg.)}
\]

\[
z_p = 2^l y_p \quad \text{(step 4 of alg.)}
\]
Thm.

\[ |(p - 1) \ln 2 + z_p| - \ln w | \leq \left( \frac{3}{4} \right)^{5\ell/2} \left( m + \frac{32}{9} + 2 \left( \frac{32}{9} + \frac{n}{\ln 2} \right)^3 \right) \]

where \( \ell \geq \lceil \log_2 8n \rceil, \ b \geq \max\{5\ell, 25\} \)

Cost:

Total # qubits is proportional to \( \ell (n + b) \log_2 b \)

Total # of quantum operations is proportional to \( \ell \) times a poly in \( n + b \)
Tests

$\sqrt{w}$: Comparison between our algorithm and Matlab

<table>
<thead>
<tr>
<th>$w$</th>
<th>Matlab: $w^{1/2}$</th>
<th>Our Algorithm: $w^{1/2}$</th>
<th># of Identical Digits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0198</td>
<td>0.140712472794703</td>
<td>0.140712472794703</td>
<td>16</td>
</tr>
<tr>
<td>48</td>
<td>6.928203230275509</td>
<td>6.928203230275507</td>
<td>15</td>
</tr>
<tr>
<td>91338</td>
<td>302.2217728754835</td>
<td>302.2217728754835</td>
<td>14</td>
</tr>
<tr>
<td>171234050</td>
<td>13085.64289593752</td>
<td>13085.64289596872</td>
<td>12</td>
</tr>
</tbody>
</table>
\( \ln w \): Comparison between our algorithm and Matlab

<table>
<thead>
<tr>
<th>( w )</th>
<th>Matlab: ( \ln(w) )</th>
<th>Our Algorithm: ( \ln(w) )</th>
<th># of Identical Digits</th>
</tr>
</thead>
<tbody>
<tr>
<td>96</td>
<td>4.564348191467836</td>
<td>4.564348191467836</td>
<td>16</td>
</tr>
<tr>
<td>65575</td>
<td>11.090949804735075</td>
<td>11.090949804735075</td>
<td>17</td>
</tr>
<tr>
<td>35711679</td>
<td>17.390988336107455</td>
<td>17.390988336107455</td>
<td>17</td>
</tr>
</tbody>
</table>
Summary

• Quantum algorithms & circuits for fundamental functions
  • Performance guarantees (accuracy, cost)

• Modular design
  • Easy to derive error bounds
  • Easy to derive resource estimates
  • Internal implementation details can be changed transparently