Lecture note 8: Quantum Algorithms

Jian-Wei Pan

Physikalisches Institut der Universität Heidelberg
Philosophenweg 12, 69120 Heidelberg, Germany
Outline

• Quantum Parallelism
• Shor’s quantum factoring algorithm
• Grover’s quantum search algorithm
Quantum Algorithm

- **Quantum Parallelism**
  - Fundamental feature of many quantum algorithms
  - It allows a quantum computer to evaluate a function $f(x)$ for many different values of $x$ simultaneously.
  - This is what makes famous quantum algorithms, such as Shor’s algorithm for factoring, or Grover’s algorithm for searching.

$$U \sum_i a_i \ket{i} = \sum_i a_i U \ket{i}$$
RSA encryption and factoring

- RSA is named after Rivest, Shamir and Adleman, who came up with the scheme
  \[ m_1 \times m_2 = N, \text{ (with } m_1 \text{ and } m_2 \text{ primes) } \]
- Based on the ease with which \( N \) can be calculated from \( m_1 \) and \( m_2 \).
- And the difficulty of calculating \( m_1 \) and \( m_2 \) from \( N \).
- \( N \) is made public available and is used to encrypt data.
- \( m_1 \) and \( m_2 \) are the secret keys which enable one to decrypt the data.
- To crack a code, a code breaker needs to factor \( N \).
RSA encryption and factoring

• Problem: given a number, what are its prime factors?
  e.g. a 129-digit odd number which is the product of two large primes,

\[11438162575788886766923577997614661201021829672124236256256184293570693524573389783059712363958705058989075147599290026879543541 = 34905295108476509491478496199038981334177646384933387843990820577 \times 32769132993266709549961988190834461413177642967992942539798288533\]

• Best factoring algorithm requires sources that grow exponentially in the size of the number
  - \( \exp(O(n^{1/3} \log^{2/3} n)) \), with \( n \) the length of \( N \)

• Difficulty of factoring is the basis of security for the RSA encryption scheme used.
Shor's algorithm

Algorithms for quantum computation: discrete logarithms and factoring


Shor, P.W.
AT&T Bell Labs., Murray Hill, NJ;
Shor’s algorithm

- **Shor’s code-breaking Quantum Algorithm**
  - How fast can you factor a number?
  - Quantum computer advantage

- Code-breaking can be done in minutes, not in millennia

- Public key encryption, based on factoring, will be vulnerable!!!

- E.g. factor a 300-digit number
  - Classical THz computer
    - $10^{24}$ steps
    - 150,000 years
  - Quantum THz computer
    - $10^{10}$ steps
    - 1 second
How to factor an odd number  
- a little number theory

• **Modular Arithmetic**

\[ a = b \mod N \iff b = a \mod N \]

simply means

\[ a = b + kN \]

where \( k \) is an integer.

• **Consider** \( x^r = 1 \mod N \)

- where \( x \) and \( N \) are co-primes, i.e. greatest common divisor \( \gcd(a,N)=1 \). No factors in common.

It will be demonstrated in the following that finding \( r \) is equivalent to factoring \( N \)
A little number theory

\[ m_1 \times m_2 = N \iff x^r = 1 \mod N \]

• Consider the equations

\[ y^2 = 1 \mod N \]
\[ y^2 - 1 = 0 \mod N \]
\[ (y + 1)(y - 1) = 0 \mod N \]
\[ (y + 1)(y - 1) = 0 \mod m_1 m_2 \]

then we have

\[
\begin{cases}
y + 1 = 0 \mod m_1 \\
y - 1 = 0 \mod m_2
\end{cases}
\text{; or }
\begin{cases}
y + 1 = 0 \mod N \\
y - 1 = 0 \mod 1
\end{cases}
\]
A little number theory

We acquire a trivial solution
\[ \begin{align*}
gcd(y + 1, N) &= N, \\
gcd(y - 1, N) &= 1. 
\end{align*} \]

and the desired solution
\[ \begin{align*}
gcd(y + 1, N) &= m_1, \\
gcd(y - 1, N) &= m_2. 
\end{align*} \]

Note that gcd can be calculated efficiently.

If we can find \( r \), and \( r \) is even
\[ y^2 = (x^{r/2})^2 = 1 \mod N \]
Then
\[ m_1 = \gcd(x^{r/2} + 1, N) \]
\[ m_2 = \gcd(x^{r/2} - 1, N) \]
provided we don’t get trivial solutions.

If \( r \) is an odd number, change \( x \), try again.
A little number theory

• Finding r is equivalent to factoring N
  - It takes $O(2^n)$ operations to find r using classical computer. (n the digits of N)

• An important result from number theory,
  \[ f(r) = x^r \mod N \]
  is a periodic function. E.g. N=15, x=7. period r= 4

• Factoring reduces to period finding.

<table>
<thead>
<tr>
<th>r</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x^r \mod N$</td>
<td>1</td>
<td>7</td>
<td>4</td>
<td>13</td>
<td>1</td>
</tr>
</tbody>
</table>
Shor algorithm

- Using quantum computer to find the period \( r \).
- The algorithm is dependent on
  - Modular Arithmetic
  - Quantum Parallelism
  - Quantum Fourier Transform

Illustration

To factor an odd integer, \( N=15 \)

Choose a random integer \( x \) satisfying \( \gcd(x,N)=1 \),
\( x=7 \) in our case.
Shor’s algorithm

- Create two quantum registers,
  - input registers: contain enough qubits to represent $r$, (8 qubits up to 255)
  - output registers: contain enough qubits to represent $f(r) = x^r \mod N < N$ (we need 4 qubits)

- Load the input registers an equally weighted superposition state of all integers (0-255). The output registers are zero.
Shor's algorithm

\[ |\psi\rangle = \frac{1}{\sqrt{256}} \sum_{a=0}^{255} |a, 0\rangle \]

\(a\) - input register, \(0\) - output register

Apply a controlled unitary transformation to the input register

\[ U |a, 0\rangle = |a, x^a \mod N\rangle \]

storing the results in the output registers.

• From quantum Parallelism, this unitary transformation can be implemented on all the states simultaneously.

\[ U |\psi\rangle = \frac{1}{\sqrt{256}} \sum_{a=0}^{255} U |a, 0\rangle = \frac{1}{\sqrt{256}} \sum_{a=0}^{255} |a, x^a \mod N\rangle \]
Shor’s algorithm

• The unitary transformation $U$ consists of a series of elementary quantum gates, single-, two-qubit...

• The sequence of these quantum gates that are applied to the quantum input depends on the classical variables $x$ and $N$ complicatedly.

• We need a classical computer processes the classical variables and produces an output that is a program for the quantum computer, i.e. the number and sequence of elementary quantum operations. This can be performed efficiently on a classical computer.

(see details, PRA, 54, 1034, (1996);)
Shor’s algorithm

Assume we applied $U$ on the quantum registers.

| in | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | ...
|----|---|---|---|---|---|---|---|---|---|---|----|----|----|---|
| out| 1 | 7 | 4 | 13| 1 | 7 | 4 | 13| 1 | 7 | 4  | 13 | 1  |...

\[
U |\psi\rangle = \frac{1}{\sqrt{256}} \left[ (|0\rangle + |4\rangle + |8\rangle + |12\rangle + \ldots |252\rangle) |1\rangle \\
+ (|1\rangle + |5\rangle + |9\rangle + |13\rangle + \ldots |253\rangle) |7\rangle \\
+ (|2\rangle + |6\rangle + |10\rangle + |14\rangle + \ldots |254\rangle) |4\rangle \\
+ (|3\rangle + |7\rangle + |11\rangle + |15\rangle + \ldots |255\rangle) |13\rangle \right]
\]

Now we measure the output registers, this will collapse the superposition state to one of the outputs $|1\rangle, |7\rangle, |4\rangle, |13\rangle$, for example $|1\rangle$. 
Shor's algorithm

- Measure the output register will collapse the input register into an equal superposition state, which is a periodic function of period $r=4$.

$$|\phi\rangle = \frac{1}{\sqrt{64}}(|0\rangle + |4\rangle + |8\rangle + |12\rangle \ldots + |252\rangle) \propto \sum_{j=0}^{M/r-1} |jr\rangle$$

- We now apply a quantum Fourier transform on the collapsed input register to increase the probability of some states.

$$T|a> = \frac{1}{\sqrt{M}} \sum_{k=0}^{M-1} e^{2\pi i \frac{ak}{M}} |k>, \quad (M=256)$$
Shor's algorithm

\[ T \left| \phi \right> \propto \sum_{j=0}^{M/r-1} T \left| j \right> = \sum_{j=0}^{M/r-1} \sum_{k=0}^{M-1} e^{2\pi i \frac{jrk}{M}} \left| k \right> \]

\[ = \sum_{k=0}^{M-1} \left( \sum_{j=0}^{M/r-1} e^{2\pi i \frac{jrk}{M}} \right) \left| k \right> = \sum_{k=0}^{M-1} f(k) \left| k \right> \]

Here \( f(k) \) can be easily calculated

\[ f(k) = \sum_{j=0}^{M/r-1} e^{2\pi i jkr/M} = \begin{cases} 
\frac{1-e^{2\pi ik}}{1-e^{2\pi ikr/M}} = 0, & kr/M \neq \text{integer} \\
M/r, & kr/M = \text{integer}
\end{cases} \]

For simplicity, we have assumed \( M/r \) is an integer
Shor’s algorithm

\[ T | \phi \rangle \propto \sum_{k=0}^{M-1} f(k) | k \rangle | 0 \rangle + \frac{1}{M} \sum_{j=0}^{r-1} | jM / r \rangle = | 0 \rangle + | 64 \rangle + | 128 \rangle + | 192 \rangle \]

- The QFT essentially peaks the probability amplitudes at integer multiples of \( M/r \). When we measure the input registers, we randomly get \( c=jM/r \), with \( 0 \leq j \leq r-1 \).

- If \( \gcd(j,r)=1 \), we can determine \( r \) by canceling \( \frac{c}{M} = \frac{j}{r} \) to an irreducible fraction.

- From number theory, the probability that a number chosen randomly from \( 1 \ldots r \) is coprime to \( r \) is greater than \( 1 / \log r \). Thus we repeat the computation \( O(\log r) \times O(\log N) \) times, we will find the period \( r \) with probability close to 1.

- This gives an efficient determination of \( r \).

(see more details in Rev. Mod. Phys., 68, 733 (1996))
Shor’s algorithm

- In our case, \( c = 0, 64, 128, 192, M = 256; \) then \( c/M = 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4} \).
- We can obtain the correct period \( r = 4 \) from \( \frac{1}{4} \) and \( \frac{3}{4} \) and incorrect period \( r = 2 \) from \( \frac{1}{2} \). The results can be easily checked from \( x^r \mod N = 1 \).
- Now that we have the period \( r = 4 \), the factors of \( N = 15 \) can be determined. This computation will be done on a classical computer.

\[
m_1 = \gcd(x^{r/2} + 1, N) = \gcd(7^{4/2} + 1, 15) = 5
\]
\[
m_2 = \gcd(x^{r/2} - 1, N) = \gcd(7^{4/2} - 1, 15) = 3
\]
Shor’s algorithm

- Generate random $x \in \{1, \ldots, N-1\}$;
- Check if $\gcd(x, N)=1$;
- $r = \text{period}(x)$;
  (The period can be evaluated in polynomial time on a quantum computer.)
- Prime factors are calculated by classical computer
  
  $m_1 = \gcd(x^{r/2} + 1, N)$
  
  $m_2 = \gcd(x^{r/2} - 1, N)$
Shor's algorithm

- $N = 15 = 5 \times 3$, the simplest meaningful instance of Shor's algorithm
- Input register 3 qubits, output register 4 qubits (Nature 414, 883, 2001)
Grover’s algorithm

- Classical search
  - sequentially try all $N$ possibilities
  - average search takes $N/2$ steps

- Quantum search
  - simultaneously try all possibilities
  - refining process reveals answer
  - average search takes $N^{1/2}$ steps

• How quickly can you find a needle in a haystack
Grover’s search algorithm

Quantum Mechanics Helps in Searching for a Needle in a Haystack

Lov K. Grover*
3C-404A Bell Labs, 600 Mountain Avenue, Murray Hill, New Jersey 07974
(Received 4 December 1996)

Quantum mechanics can speed up a range of search applications over unsorted data. For example, imagine a phone directory containing $N$ names arranged in completely random order. To find someone’s phone number with a probability of 50%, any classical algorithm (whether deterministic or probabilistic) will need to access the database a minimum of $0.5N$ times. Quantum mechanical systems can be in a superposition of states and simultaneously examine multiple names. By properly adjusting the phases of various operations, successful computations reinforce each other while others interfere randomly. As a result, the desired phone number can be obtained in only $O(\sqrt{N})$ accesses to the database.

[S0031-9007(97)03564-3]

**Grover’s search algorithm**

- Problem: given a Quantum oracle, 1, 2, ..., x, ..., N try to find one specific state $x$, satisfying $R$ is a $N \times N$ diagonal matrix, satisfying $R_{ii} = -1$, if $i = x$; $R_{ii} = 1$, other diagonal elements. To find $x$ is equivalent to find which diagonal element of $R$ is -1, i.e. $x$.

- Classically, we have to go through every diagonal element. We expect to find the -1 term after $N/2$ queries to all the diagonal elements.

$$R |i\rangle = \begin{cases} -|i\rangle, & i = x \\ |i\rangle, & \text{otherwise} \end{cases}$$
Grover’s algorithm

• Take a m-qubit register, assume $2^m = N$
• Prepare the registers in an equal superposition state of all the $N$ states.

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle$$

• Iterations of Rotate Phase and Diffusion operator
• Measure the register to get the specific state $x$
Grover’s algorithm

- In fact, $R$ is a phase rotate operator

$$R|i\rangle = \begin{cases} 
-|i\rangle, & i = x \\
|i\rangle, & \text{otherwise}
\end{cases}$$

e.g. $x = 1$

$$R = \begin{bmatrix}
1 & 0 & 0 & \cdots & 0 \\
0 & -1 & 0 & \cdots & 0 \\
0 & 0 & 1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1
\end{bmatrix}$$
Grover's algorithm

- Diffusion operator

$$D = \begin{bmatrix}
-1 + \frac{2}{N} & \frac{2}{N} & \cdots & \frac{2}{N} \\
\frac{2}{N} & -1 + \frac{2}{N} & \cdots & \frac{2}{N} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{2}{N} & \frac{2}{N} & \cdots & -1 + \frac{2}{N}
\end{bmatrix}$$

The successive operation of Rotate phase and Diffusion operator will increase the probability amplitude of the desired state.
Grover’s algorithm

• Initial state
  \[ |\psi\rangle = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle = \frac{1}{\sqrt{N}} |x\rangle + \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle = |\alpha\rangle + |\beta\rangle \]

  \[ |\psi_n\rangle = (DR)^n |\psi\rangle, \quad \begin{cases} \hat{D}R |\alpha\rangle = \left(1 - \frac{2}{N}\right)|\alpha\rangle - \frac{2}{N}|\beta\rangle \\ \hat{D}R |\beta\rangle = \left(2 - \frac{2}{N}\right)|\alpha\rangle + \left(1 - \frac{2}{N}\right)|\beta\rangle \end{cases} \]

• After n iteration, we have
  \[ |\psi_n\rangle = a_n |\alpha\rangle + b_n |\beta\rangle, \quad \begin{pmatrix} a_n \\ b_n \end{pmatrix} = \begin{pmatrix} 1 - \frac{2}{N} & 2 - \varepsilon \\ -\varepsilon & 1 - \varepsilon \end{pmatrix} \begin{pmatrix} a_{n-1} \\ b_{n-1} \end{pmatrix}, \text{ with } \begin{cases} a_0 = 1 \\ b_0 = 1 \end{cases} \]

• Considering \( N \not\equiv 1 \)
Grover’s algorithm

- Finally, we get

\[ |\psi_n\rangle \approx \sin \left( \frac{2n}{\sqrt{N}} \right) |x\rangle + \frac{\cos \left( \frac{2n}{\sqrt{N}} \right)}{\sqrt{N}} \sum_{i=0 \atop (i \neq x)}^{N-1} |i\rangle \]

- The probability to collapse into the \( x \)

\[ P|n\rangle = \left| \sin \left( \frac{2n}{\sqrt{N}} \right) \right|^2 \]

- We choose iteration steps \( n = \left[ \frac{\pi}{4} \sqrt{N} \right] \)
the probability of failure

\[ 1 - p(n) \leq \cos^2 \left( \frac{\pi}{2} - \frac{1}{\sqrt{N}} \right) = O \left( \frac{1}{N} \right) \xrightarrow{N \to \infty} 0 \]
Grover’s algorithm

- Can we do better than a quadratic speed up for Quantum Searches.
- No! Grover algorithm is optimal. Any quantum algorithm, with respect to an Oracle, can not do better that Quadratic time.
- Good and Bad
  - Good: Grover’s is Optimal
  - Bad: No logarithmic time algorithm

:Limits of “Black-Box” quantum computing
Grover’s algorithm

- Experiment realization
  - Nuclear magnetic resonance
  - Linear optics
  - individual atom
  - trapped ion
    M. Feng, PRA, 63, 052308 (2001).