Introduction to Quantum Information Processing

CS 467 / CS 667
Phys 467 / Phys 767
C&O 481 / C&O 681

Lecture 8 (2005)

Richard Cleve
DC 3524
cleve@cs.uwaterloo.ca

Course web site at:
http://www.cs.uwaterloo.ca/~cleve

Contents

- Recap of the order-finding problem/algorithm
- Reduction from factoring to order-finding
- The discrete log problem
- The “hidden subgroup” framework

- Recap of the order-finding problem/algorithm
- Reduction from factoring to order-finding
- The discrete log problem
- The “hidden subgroup” framework

Order-finding problem

Input: M (an n-bit integer) and $a \in \{1,2,\ldots, M-1\}$ such that $\mathrm{gcd}(x, M) = 1\}$

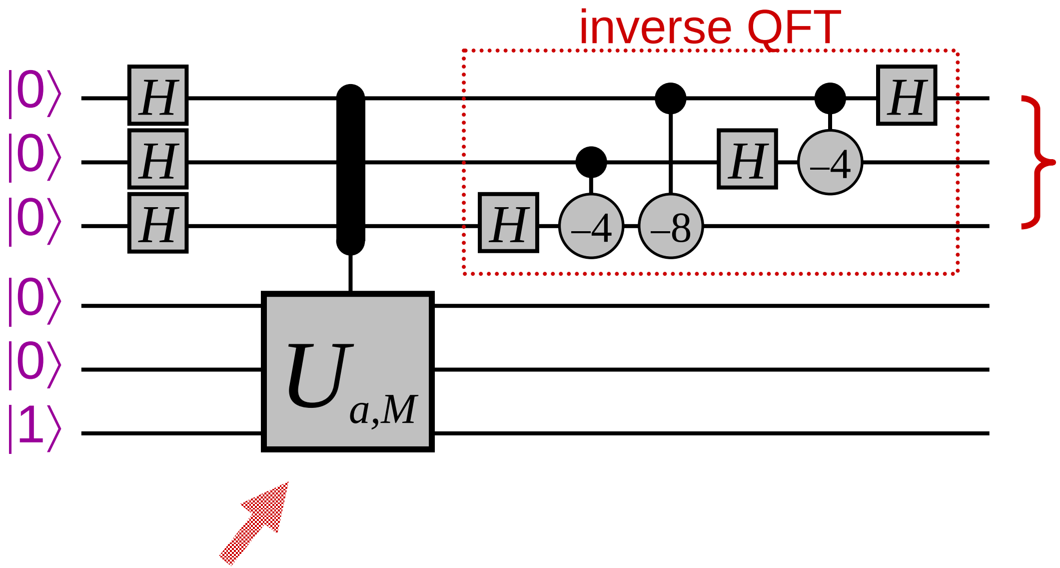
Output: $\mathrm{ord}_M(a)$, which is the minimum $r > 0$ such that $a^r = 1 \ (\mathrm{mod}\ M)$

Example: for $M = 21$ and $a = 6$, the powers of 10 are:
$1, 10, 16, 13, 4, 19, 1, 10, 16, 13, 4, 19, 1, 10, 16, 13, 4, \ldots$

Therefore, the correct output is: 6

Note: no ***classical*** polynomial-time algorithm is known for this problem

Quantum algorithm for order-finding



measure these qubits and apply continued fractions* algorithm to determine a quotient, whose denominator divides *r*

$$U_{a,M}|y\rangle = |ay \bmod M\rangle$$

Number of gates for a constant success probability is: $O(n^2 \log n \operatorname{loglog} n)$

* For a discussion of the *continued fractions algorithm*, please see Appendix A4.4 in [Nielsen & Chuang]

- Recap of the order-finding problem/algorithm
- Reduction from factoring to order-finding
- The discrete log problem
- The “hidden subgroup” framework

The integer factorization problem

Input: M (n-bit integer; we can assume it is composite)

Output: p, q (each greater than 1) such that $pq = N$

Note 1: no efficient (polynomial-time) classical algorithm is known for this problem

Note 2: given any efficient algorithm for the above, we can recursively apply it to fully factor M into primes* efficiently

* A polynomial-time ***classical*** algorithm for ***primality testing*** exists

Factoring prime-powers

There is a straightforward ***classical*** algorithm for factoring numbers of the form $M = p^k$, for some prime p

What is this algorithm?

Therefore, the interesting remaining case is where M has at least two distinct prime factors

Numbers other than prime-powers

Proposed quantum algorithm (repeatedly do):

1. randomly choose $a \in \{2, 3, \ldots, M-1\}$
2. compute $g = \gcd(a, M)$
3. **if** $g > 1$ **then**

    output g, M/g

  else

    compute $r = \mathrm{ord}_M(a)$ (quantum part)

    if r is even **then**

      compute $x = a^{r/2} - 1 \bmod M$

      compute $h = \gcd(x, M)$

      if $h > 1$ **then** output h, M/h

Analysis:

we have $M \mid a^r - 1$

so $M \mid (a^{r/2}+1)(a^{r/2}-1)$

thus, either $M \mid a^{r/2}+1$
or $\gcd(a^{r/2}+1, M)$
is a nontrivial factor of M

latter event occurs with probability $\geq$ ½

- Recap of the order-finding problem/algorithm
- Reduction from factoring to order-finding
- The discrete log problem
- The “hidden subgroup” framework

Discrete logarithm problem (I)

Input: p (prime), g (generator of $\mathbf{Z}^*_p$), $a \in \mathbf{Z}^*_p$

Output: $r \in \mathbf{Z}_{p-1}$ such that $g^r \bmod p = a$

Example: $p = 7$, $\mathbf{Z}^*_7 = \{1, 2, 3, 4, 5, 6\} = \{3^0, 3^2, 3^1, 3^4, 3^5, 3^3\}$ (hence 3 is a generator of $\mathbf{Z}^*_7$)

For $a = 6$, since $3^3 = 6$, the output should be $r = 3$

Note: No efficient classical algorithm for ***DLP*** is known (and cryptosystems exist whose security is predicated on the computational difficulty of DLP)

Efficient quantum algorithm for DLP?
(**Hint:** it can be made to look like Simon's problem!)

Discrete logarithm problem (II)

Clever idea (of Shor)**:** define $f : \mathbf{Z}_{p-1} \times \mathbf{Z}_{p-1} \rightarrow \mathbf{Z}^*_p$ as $f(x, y) = g^x a^{-y} \bmod p$

When is $f(x_1, y_1) = f(x_2, y_2)$?

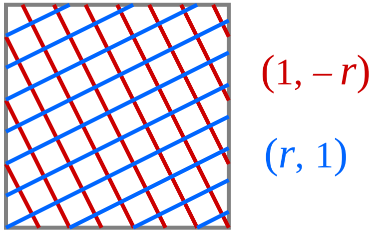
We know $a = g^r$ for ***some*** r, so $f(x, y) = g^{x - ry} \bmod p$

Thus, $f(x_1, y_1) = f(x_2, y_2)$ iff $x_1 - ry_1 \equiv x_2 - ry_2 \pmod{p-1}$

iff $(x_1, y_1) \cdot (1, -r) \equiv (x_2, y_2) \cdot (1, -r) \pmod{p-1}$

iff $((x_1, y_1) - (x_2, y_2)) \cdot (1, -r) \equiv 0 \pmod{p-1}$

iff $(x_1, y_1) - (x_2, y_2) \equiv k(r, 1) \pmod{p-1}$



$\mathbf{Z}_{p-1} \times \mathbf{Z}_{p-1}$

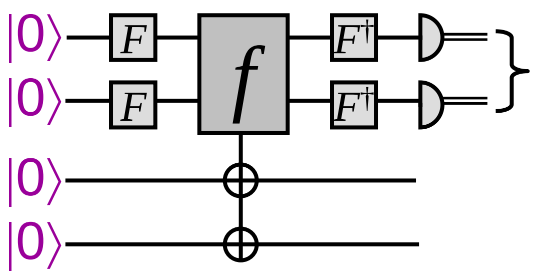
Recall Simon's: $f(x) = f(y)$ iff $x - y = kr \pmod{2}$

Discrete logarithm problem (III)

$f : \mathbf{Z}_{p-1} \times \mathbf{Z}_{p-1} \rightarrow \mathbf{Z}^*_p$ defined as $f(x, y) = g^x a^{-y} \bmod p$

$$f(x_1, y_1) = f(x_2, y_2) \text{ iff } (x_1, y_1) - (x_2, y_2) \equiv k(r, 1) \pmod{p-1}$$

Recall Simon's: $f(x) = f(y)$ iff $x - y = kr \pmod 2$



result is a random (s, t) such that

$(s, t) \cdot (r, 1) \equiv 0 \pmod{p-1}$ **Why?**

if $\gcd(s, p-1) = 1$ then r can be computed as $r = -ts^{-1} \bmod p-1$

- Recap of the order-finding problem/algorithm
- Reduction from factoring to order-finding
- The discrete log problem
- The “hidden subgroup” framework

Hidden subgroup problem (I)

Let G be a known group and H be an unknown subgroup of G

Let $f: G \rightarrow T$ have the property $f(x_1) = f(x_2)$ iff $x_1 - x_2 \in H$ (i.e., x_1 and x_2 are in the same ***right coset*** of H)

Problem: given a black-box for computing f, determine H

Example 1: $G = (\mathbf{Z}_2)^n$ (the additive group) and $H = \{0, r\}$

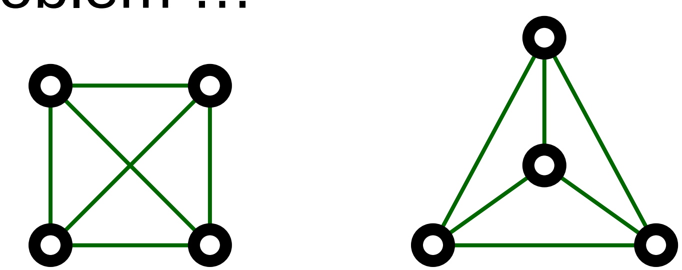
Example 2: $G = (\mathbf{Z}_{p-1})^2$ and
$H = \{(0,0), (r,1), (2r,2), \ldots, ((p-2)r, p-2)\}$

Example 3: $G = \mathbf{Z}$ and $H = r\mathbf{Z}$

Hidden subgroup problem (II)

Example 4: $G = S_n$ (the symmetric group, consisting of all permutations on n objects—which is not abelian) and H is any subgroup of G

A quantum algorithm for this instance of HSP would lead to an efficient quantum algorithm for the graph isomorphism problem …



… yet no efficient quantum has been found for this instance of HSP, despite significant effort by many people

THE END