CS 880: Quantum Algorithms 3/25/2021

Lecture 17: Integer Factorization and Discrete Log

Instructor: Dieter van Melkebeek Scribe: Matthew Wallace

This lecture is about challenges. In the context of this lecture, "challenges" , describe problems posed by the existence of quantum computing to systems using classical cryptography. A number of quantum algorithms, which are described below, have the potential to break both factoring-based and discrete-log-based cryptosystems. This class also demonstrates that information-theoretic security is impossible in the quantum setting. Essentially, this means that quantum computing systems are not able to produce a plaintext that can only be deduced if the key is known. This occurs because perfect bit commitment is not possible (discussed in section 6).

1 Integer Factorization

Problem

- Input: positive integer, μ
- Output: factorization of μ into primes

Complexity as a function of $n = \log \mu$

- Classical rigorous complexity: $2^{\widetilde{O}(n^{1/2})}$
- Classical heuristic complexity: $2^{\widetilde{O}(n^{1/3})}$
- Quantum complexity: $\widetilde{O}(n^2)$

What is the difference between rigorous complexity and heuristic complexity?

- Rigorous complexity implies an exact solution with a proven bound.
- Heuristic complexity implies that the solution employed lacks a concrete proof for the suspected runtime complexity.

Why is the heuristic algorithm used for integer factorization in this case? In the context of integer factorization, the heuristic listed above is used to gauge a ballpark estimate for how long your keys need to be when employing cryptosystems such as RSA. Later on in section 2, we discuss RSA in more detail, as it is a key system that is commonly used in cryptography applications.

How is integer factorization solved in a classical setting? In order to represent the problem in a way that is solvable by a quantum computing system, we reduce the integer factorization problem to a splitting problem. Then, we reduce the splitting problem to an order finding problem.

What is splitting? Given a collection of subsets, B, of a finite set, A, the splitting problem is to decide whether there exists a partition of A into two subsets A1, A2 such that all elements of B are split by the chosen partition. In other words, for each subset in the collection which forms B, no subset is contained wholly in A1 or A2, which is the partition of A.

What is order finding?

- Input: $a, \mu \in \mathbb{N}$ such that $\gcd(a, \mu) = 1$
- Output: smallest positive $r \in \mathbb{N}$ such that $a^r = 1 \bmod \mu$

Next, we introduce two new lemmas (1 and 2) that show how splitting can be reduced to order finding.

Lemma 1. *If b^2 equals $1 \bmod \mu$ and b is not equal to $1 \bmod \mu$ and b is not equal to $-1 \bmod \mu$; then, the greatest common denominators of $b-1$ and μ and $b+1$ and μ are nontrivial factors of μ.*

In Symbols: $(b^2 = 1 \bmod \mu) \wedge (b \neq 1 \bmod \mu) \wedge (b \neq -1 \bmod \mu) \rightarrow \gcd(b-1, \mu) \wedge \gcd(b+1, \mu)$ *are nontrivial factors of μ*

Proof. Neither $b \pm 1$ is a multiple of μ because $(b - 1 \neq 0 \bmod \mu) \wedge (b + 1 \neq 0 \bmod \mu)$. But, $(b-1)(b+1) = b^2 - 1$ is a multiple of μ, which implies that $b \pm 1$ are nontrivial factors of μ. □

How Do We Use Lemma 1 to Find b?

Lemma 2. *Suppose at least j distinct primes divide μ. For a fraction at least $1 - \frac{1}{2^j}$ of $a \in \mathbb{Z}_\mu^x$, the order r of a is even, and $b = a^{r/2} \bmod \mu$ satisfies $(b \neq 1 \bmod \mu) \wedge (b \neq -1 \bmod \mu)$.*

If we compute $a^{\frac{r}{2}} \bmod \mu$ (this is well defined because r is even), for a fraction at least $1 - \frac{1}{2^j}$ the following is true: the order of $ra \bmod u$ is even and moreover, if we look at b, we can claim that it has the property of lemma 1. This can be done because lemma 2 states that $b \neq 1 \bmod \mu$ and $b \neq -1 \bmod \mu$, which implies that $b - 1$ and $b + 1$ are nontrivial factors of μ.

$b^2 = 1 \bmod \mu$ has the first power automatically, and moreover, for the fraction, r is even and b is not equal to $1 \bmod \mu$ or $-1 \bmod \mu$. Since we can get b from lemma 2, we can then use b to compute a nontrivial factor of μ.

Notes on Lemma 2 For $j = 1$, the statement holds vacuously. The proof of this lemma uses the Chinese Remainder Theorem, but doesn't have much to do with class material, so it is omitted.

Integer Factorization Algorithm

Complexity: Polynomial in the bitlength of μ. Polylogarithmic in μ itself.

Description

1. Check whether μ is a prime. If yes, return μ.
2. If $\mu = (\mu^{'})^k$ for some integers $(\mu^{'})$ and $(k > 1)$; then, recursively factor $\mu^{'}$ and repeat the factorization k-times
3. Pick a random $a \in \{1, 2, ..., \mu - 1\}$.

   If $\gcd(a, \mu) \neq 1$ then $\mu^{'} := \gcd(a, \mu)$. Else, compute the order r of $a \bmod \mu$.

   If r is even and $b = a^{r/2} \bmod \mu$ satisfies $(b \neq 1 \bmod \mu) \wedge (b \neq -1 \bmod \mu)$; then, $\mu^{'} := \gcd(b + 1, \mu)$

   Else, try again with a new a value.
4. Recursively factor $\mu^{'}$ and $\frac{\mu}{\mu^{'}}$

Probability of Success (after k trials): $1 - \frac{1}{2^k}$

Analysis The probability of success of the first if branch is 1. The probability of success of the first else branch in the integer factorization algorithm is at least $\frac{1}{2}$. The expected number of tries necessary for success is at most 2. Probability of executing for more than k iterations before finding successful a is $\frac{1}{2^k}$, an exponential decrease.

Why do we call the nontrivial factor $\mu^{'}$? Because once we have $\mu^{'}$, we perform splitting again by factoring $\mu^{'}$ and μ / $\mu^{'}$.

Note: There exists no classical algorithm to compute the order r of $a \bmod \mu$ efficiently.

Why is this algorithm important? The hardness of factoring integers is used prominently throughout many different cryptography systems. Being able to factor integers quickly breaks many commonly-used schemes, like RSA, which is usually used to secure e-commerce transactions.

2 RSA Crypto

RSA is a public-key cryptosystem wherein every participating entity has a publicly-viewable encryption key and a private decryption key that only they know. This protocol relies on the hardness of factoring for security. The acronym is composed of the first letter of each of the three computer scientists who developed the protocol, Rivest, Shamir, and Adleman.

Complexity: Polynomial in the bitlength of μ.

Description

- Private Key

  two distinct primes, p and q

  $d \in \mathbb{N}$ relatively prime with $(p-1)$ and $(q-1)$

- Public Key

  $\mu = p \cdot q$

  $e \in \mathbb{N}$ such that $de = 1 \bmod (p-1)(q-1)$

Protocol

1. Before the protocol begins, the entities planning to use RSA must share a public key. For example, if Bob wants to send a message to Alice, he needs to know her public key in order to encrypt his message. Alice will need her private key in order to decrypt the message sent from Bob. Note that the public key does not need to be sent secretely for this protocol to work as intended.

2. $M \in \mathbb{Z}_\mu \rightarrow E = M^e \bmod \mu$. In this step, Bob encrypts his message and turns it into an integer, M, such that $0 \leq M < \mu$ using Alice's public key, e.

3. $M^e \bmod \mu \rightarrow D = E^d \bmod \mu = M^{ed} \bmod \mu = M$. This part of the protocol corresponds with Alice's decryption of Bob's message. Alice recovers Bob's message, M, by using her private key exponent, d, to compute $M^{ed} \bmod \mu$.

Proof of Correct Decryption:

Proof. If $\gcd(M,p) = 1$ then $M^{ed} = M^{1+k(p-1)} = M \bmod p$. If $\gcd(M,p) \neq 1$ then $M^{ed} = M^{1+k(p-1)} = 0 = M \bmod p$. So, $M^{ed} = M \bmod p$. The same holds for mod$q$ and thus also for mod μ. The intuition here is that when we want to check if M^{ed} and M are congruent mod pq, it is sufficient to check for congruent with respect to mod$p$ and mod$q$ separately. If we can show this, as is done above, we know that $M^{ed} = m \bmod pq$. □

RSA construction note: The private key consists of two distinct primes, which are picked at random to prevent an attacker from gaining any unnecessary advantage when trying to guess p and q in order to replicate the private key.

Vulnerability: Integer factorization enables a potential attacker to retrieve a private key from its corresponding public key. Classically, it is an open question as to whether RSA is secure. AKA, it is not known whether breaking RSA is equivalent to Integer Factorization. Integer factorization implies the ability to break RSA, but solving RSA does not necessarily imply the ability to factor integers quickly.

3 The Discrete Log Problem

Problem

- For prime, p, multiplicative group $\mathbb{Z}_p^\times = \{x \in \mathbb{Z}_p : \gcd(x, p) = 1\} = \{1, 2, ...\}$ is cyclic
- Input: a prime p and a generator g for $\mathbb{Z}_p^\times, a \in \mathbb{Z}_p$
- Output: $l \in \mathbb{Z}_{p-1}$ such that $g^l = a \bmod \mu$

Complexity as a function of $n = \log p$:

- Classical rigorous complexity: $2^{\widetilde{O}(n^{1/2})}$
- Classical heuristic complexity: $2^{\widetilde{O}(n^{1/3})}$
- Quantum complexity: $\widetilde{O}(n^2)$

Comparison to Shor's Algorithm Shor's algorithm runs in time that is polynomial to log n, where n is the inputted integer. Shor's algorithm is composed of five steps, only one of which needs the use of a quantum computer. The step that requires a quantum computer deals with finding a period of, p, of the function $f(p) = m^p \bmod n$. Classical systems need super-polynomial time in the number of digits in the input, n.

Problem Formulation Note: We solve the discrete log problem as an instantiation of the hidden subgroup problem. In general, it is possible to efficiently solve the hidden subgroup problem over finite abelian groups, provided that you know the decomposition of the group.

4 Discrete Log Cryptography

The Diffie-Hellman protocol leverages the difficulty of the discrete log problem, described in the prior section, to develop a public-key cryptosystem. D-H allows two parties to communicate over an insecure communication channel by creating a shared secret key. The shared key that is created is used in subsequent messages sent between the two communicating parties. This method is known as a symmetric cryptography system, given that both parties develop one key. This system was developed by Whitfield Diffie and Martin Hellman in the late 1970s.

Diffie-Hellman key exchange

- Public Key is composed of:

  a prime p

  and a generator g for $\mathbb{Z}_p^\times$

- D-H Protocol:

  Alice picks $a \in \mathbb{Z}_p^\times$ at random, sends $A = g^a \bmod p$ to Bob

  Bob picks $\mathbb{Z}_p^\times$ at random, sends $B = g^b \bmod p$ to Alice

Alice computes $K_A = B^a \bmod p$

Bob computes $K_B = A^b \bmod p$

Alice and Bob use $K_A = g^{ab} \bmod p = K_B$ as key

Important fact regarding D-H scheme: If x is random, g^x is randomly distributed.

Vulnerability: A fundamental vulnerability of the Diffie-Hellman scheme is its reliance on the discrete log problem being difficult to solve. If one can solve discrete log, one can retrieve private choices a and b from transmitted values! Here's how: suppose Alice sends her message A to Bob. The adversaries can take the transmitted A value and use their discrete log subroutine to find the discrete log, a. The adversaries can also do this for Bob's message, b. Once the adversaries have both, the protocol can be simulated. It is sufficient to use either a or b to find the shared key from intercepted communication; this will not require the computation of two discrete logs!

Vulnerability side note: The vulnerability described above follows because breaking D-H is an instance of the hidden subgroup problem over a finite Abelian group with known decomposition.

Side note: Elliptic curve crypto can also be broken quantumly.

5 Quantum and Crypto

Challenges: All cryptosystems that use the hardness of integer factorization, discrete log, or elliptic curve problems can be compromised.

Classical systems and information-theoretic security: Information-theoretic security is not possible in the classical setting.

- Classical cryptosystems that remain secure in the quantum setting.

  Example: Certain Zero-Knowledge systems

  Exploit: Quantum alternatives to do better than in the classical setting.

- Are information-theoretically secure key exchanges possible in a quantum computing system? Neither Diffie-Hellman nor RSA is information-theoretically secure.

Opportunities:

The largest number factored by a quantum computer to date is 21. We are currently not very close to being able to factor integers that are too large for classical computers to factor. Factoring and discrete log are NP-Intermediate problems, so cryptographic systems that rely on a more difficult set of problems on the complexity heirarchy could produce more security.

Research question: Are lattice-based cryptosystems secure in a quantum setting?

Research Question: Design a quantum cryptosystem that uses NP-hard problems.

Research Question: Are NP-Hard problems computable in polynomial time by quantum computers? The most common thinking on this currently is that NP-Hard problems are not computable in polynomial time.

6 Bit Commitment

Definition

- Two stage protocol:
  1. Commit: Alice sends info to Bob that commits her to a bit b.
  2. Reveal: Alice releases more info (the key) that lets Bob figure out the value of b.

Requirements of bit commitment

1. Hiding: Bob learns nothing about the value of b during the commit phase
2. Binding: Alice cannot change the value of b after the commit phase

Note that these requirements are information-theoretic and computationally secure.

Problem: There exists no bit commitment scheme of information-theoretic strength in the quantum setting. Why is this the case? To show why, we can use the reduced density operator.

Schmidt Decomposition

Lemma 3. *Given a state $|\psi_{AB}\rangle$, there exist orthonormal bases, $\{|\phi_{A,i}\rangle\}_i$ for Alice's part of the state, and $\{|\phi_{B,i}\rangle\}_i$ for Bob's part of the state, and nonnegative reals, λ_i such that $|\psi_{AB}\rangle = \Sigma_i \lambda_i |\phi_{A,i}\rangle |\phi_{B,i}\rangle$.*

Proof. Follows directly from SVD:

$$\begin{pmatrix} |\psi_{A,0}\rangle & ... & |\psi_{A,3}\rangle & ... \end{pmatrix} \begin{pmatrix} \lambda_0 & 0 & 0 & 0 \\ 0 & \lambda_1 & 0 & 0 \\ 0 & 0 & ... & 0 \\ 0 & 0 & 0 & \lambda_n \end{pmatrix} \begin{pmatrix} |\psi_{B,0}\rangle \\ ... \\ ... \\ |\psi_{B,n}\rangle \end{pmatrix}$$

□

Corollary 4. *$\rho_A = \Sigma_i \lambda_i^2 |\psi_{A,i}\rangle \langle\psi_{A,i}|$ and $\rho_B = \Sigma_i \lambda_i^2 |\psi_{B,i}\rangle \langle\psi_{B,i}|$. Explanation: This corollary follows because the schmidt decomposition of AB is given as $|\psi_{AB}\rangle = \Sigma_i \lambda_i |\psi_{A,i}\rangle |\psi_{B,i}\rangle$. The density matrix of AB is $\rho = \Sigma_i |\psi\rangle \langle\psi| = \Sigma_i \lambda_i^2 |\psi_{A,i}\rangle \langle\psi_{A,i}| \otimes |\psi_{B,i}\rangle \langle\psi_{B,i}|$ This equation can be decomposed to produce ρ_A and ρ_B.*

Perfect bit commitment is impossible:

- Suppose there is a protocol that is perfectly hiding
- Consider purifications of states of the system after commitment phase: $|\psi^b_{AB}\rangle$ for $b \in \{0, 1\}$.
- Their Schmidt Decompositions would be: $|\psi^b_{AB}\rangle = \Sigma_i \lambda^b_i |\psi^b_{A_i}\rangle |\psi^b_{B_i}\rangle$
- Since $\rho^0_B = \rho^1_B$ *and* $\rho^B_b = \Sigma_i (\lambda^{(b)}_i)^2 |\psi^{(b)}_{B_i}\rangle \langle\psi^{(b)}_{B_i}|$ the following are true:
  1. $\lambda^{(0)}_i = \lambda^{(1)}_i$
  2. $|\psi^{(0)}_{B_i}\rangle = |\psi^{(1)}_{B_i}\rangle, \forall i$
  3. Statements 1 and 2 imply that $|\psi^{(b)}_{AB}\rangle = \Sigma_i \lambda_i |\phi^{(b)}_{A_i}\rangle |\phi^{(b)}_{B_i}\rangle$ (Note that superscripts disappeared for $|\phi_{B,i}\rangle$ and λ_i)

Intuition on why these properties show impossibility of perfect bit commitment:

Binding Density operators must be the same regardless of bit choice, as shown in subitem 3 immediately above. Recall that density operators are hermitian, which implies that their eigenvalues are the same. There exists a Unitary matrix U such that $(U \times I) |\psi^{(0)}_{AB}\rangle = |\psi^{(1)}_{AB}\rangle$. This breaks the binding requirement because Alice can 'change' her bit choice after the fact, which Bob will be unable to detect because density operators are the same regardless of the bit Alice chooses.

Therefore, if a protocol is perfectly hiding, it is not perfectly binding. This impossibility means that information-theoretic secure protocols are impossible in quantum systems, just like they are with classical systems!

Next time: We cover post-quantum cryptography and positive opportunities in the space. We will also dive into interactive proofs, which provide useful background for the course project.