Hidden Subgroup Problem

Karn Vadaliya	201501116
Raj Gamit	201501171
Harsh Makwana	201501191
Priyam Suthar	201501249

What is it ?

Given a group G, a subgroup $H \leq G$, and a set X, we say a function $f : G \to X$ **hides** the subgroup H if for all $g_1, g_2 \in G$, $f(g_1) = f(g_2)$ if and only if $g_1H = g_2H$ for the cosets of H. Equivalently, the function f is constant on the cosets of H, while it is different between the different cosets of H.

Why do we care ?

- Shor's quantum algorithm for factoring and discrete logarithm (as well as several of its extensions) relies on the ability of quantum computers to solve the HSP for finite Abelian groups.
- The existence of efficient quantum algorithms for HSPs for certain non-Abelian groups would imply efficient quantum algorithms for two major problems: the graph isomorphism problem and certain shortest vector problems (SVPs) in lattices. More precisely, an efficient quantum algorithm for the HSP for the symmetric group would give a quantum algorithm for the graph isomorphism. An efficient quantum algorithm for the HSP for the dihedral group would give a quantum algorithm for the poly(n) unique SVP.

Some Algorithms under HSP

For Abelian groups

- Bernstein-Vazirani - F:$\{0,1\}^n \rightarrow$ F:$\{0,1\}$, where F(x)=X.S, H={z belongs to Z_2^n | X.z=0 }
- Simon's Problem - F:$\{0,1\}^n \rightarrow$ G (some set), F(x) =F(x xor L) "L shift" for every X, FindL.
- Period Finding - $Z_n \rightarrow$ G(Some set) , F(x)= F(x+nLmod N), n belongs to {0,1,2...}, L divides N and it is the period, Find L.
- Shor Algorithm

For non-abelian groups

- Dihedral group D_n - G= {all permutation π: {1,2..n} → {1,2..n} that are automorphisms of N-cycle graph, can be applied in approximate shortest vector problem in lattice (NP-hard)
- Symmetric group S_n- G={ all permutation π: {1,2..n} → {1,2..n}} for Graph isomorphism

Complexity

Both query and time complexities for quantum algorithm are polynomial in log(|G|), which is significantly smaller than classical complexities.

Shor’s Algorithm

- Shor’s algorithm shows that a quantum computer is capable of factoring very large numbers in polynomial time
- The problem is: given an odd composite number N, find and integer d, strictly between 1 and N, that divides N.
- The algorithm is dependent on
    - Modular Arithmetic
    - Quantum Parallelism
    - Quantum Fourier Transform
- Finding a factor of a n-bit integer requires exp(e(n^⅓(log n)^⅔) operations using classical algorithm but Shor’s algorithm can accomplish this task in O(n^2(log n(log log n)) operations.

Shor’s Algorithm

Shor's algorithm consists of two parts:

- A reduction, which can be done on a classical computer, of the factoring problem to the problem of order-finding. (Classical Part)
- A quantum algorithm to solve the order-finding problem. (Quantum Part - Period Finding subroutine)

Classical Part

1. A random number $a<N$ is picked.
2. Compute gcd(a, N). This may be done using the Euclidean algorithm.
3. If $\gcd(a,N) \neq 1$, then there is a nontrivial factor of N.
4. $F(x+r)=a^{x+r} \bmod N = a^{x} \bmod N = f(x)$.
5. If r is odd, go to step 1.
6. If $a^{r/2} = -1 \pmod N$, go back to step 1.
7. $\gcd(a^{r/2} \pm 1, N)$ is non trivial factor of N.

Depth Analysis

To Factor an odd integer N (Let's choose N=15) :

1. Choose an integer q such that $N^2 < q < 2N^2$; let's pick 256
2. Choose a random integer x such that GCD(x, N) = 1 ; let's pick 7
3. Create two quantum registers (these registers must also be entangled so that the collapse of the input register corresponds to the collapse of the output register)
   a. Input register must contain enough qubits to represent numbers as large as q-1. ; Up to 255, so we need 8 qubits
   b. Output register: must contain enough qubits to represent numbers as large as N-1. ; Up to 14, so we need 4 qubits.

4. Load the input register with an equally weighted superposition of all integers from 0 to q-1. 0-255

5. Load the output register with all zeros.

The total state of the system at this point will be[redacted]

6. Apply the transformation x^a mod N to each number in the input register.storing the result of each computation in the output register.

7. Now take a measurement on the output register. This will collapse the superposition to represent just one of the results of the transformation, let's call this value c.

8. Since the two registers are entangle, measuring the output register will have effect of partially collapsing the input register into an equal superposition of each state between 0 and q-1 that yielded c (the value of the collapsed output register.)

9. We now apply the Quantum Fourier transform on the partially collapsed input register The Fourier transform on the partially collapsed input register. The Fourier transform has effect of taking a state |a> and transforming it into a state given by : $\frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} |c> * e^{2\pi iac / q}$

10. Now that we have the period, the factors of N can be determined by taking the greatest common divisor of N with respect to x^(p/2) + 1 and

x^(p/2) - 1. The idea here is that this computation will be done on a classical computer.

Shor's Algorithm-Problems

- The QFT comes up short and reveals the wrong period. This probability is actually dependant on your choice of q. The larger the q, the higher the probability of finding the correct probability.
- The period of the series ends up being odd.
    - If either of these cases occur, we go back to the beginning and pick a new x.
- Quantum modular exponentiation, much slower than the quantum Fourier transform.

Applications

- Factoring RSA - RSA is based on assumption that factoring large numbers is infeasible.
- Quantum Simulation - Quantum simulator permit the study of quantum systems that are difficult to study in the laboratory and impossible to model with a supercomputer.
- Spin-off technology - spintronics, quantum cryptography

Thank You