Contents

1 Order finding
   - Order finding - the problem
   - Order finding algorithm

2 Discrete log
   - Discrete log - the problem
   - Discrete log - the algorithm

3 The HSP
   - Common features of order finding and discrete log
   - Generalizations
   - The HSP
   - The Graph isomorphism problem
Given $u$ in a group (say, $u \in \mathbb{Z}_N^*$). Find the (multiplicative) order of $u$.

Useful in factoring integers:

- $N$: a composite odd number
- Pick random $x \in \mathbb{Z}_N \setminus \{0\}$. With probability $> \text{constant/} \log \log N$), $x \in \mathbb{Z}_N^*$ such that
  - $y^2 = 1$, but $y \neq \pm 1$,
    where $y$ is the smallest power of $x$ s.t. $y^2 = 1$.
  - Either for $z = y + 1$ or for $z = y - 1$: $0 \neq z \in \mathbb{Z}_N \setminus \mathbb{Z}_N^*$
  - $\gcd(x, N)$ is a proper divisor of $N$

Here a much weaker version than Shor's, we assume the a multiple of the order is known:

Given $u$ in a group (say, $u \in \mathbb{Z}_N^*$) and $n \in \mathbb{Z}_{>0}$ s.t. $u^n = 1$. Find the order of $u$. 
Order finding algorithm 1.

1. \( \frac{1}{\sqrt{n}} \sum_{i=0}^{n-1} |i\rangle|1\rangle \)

   Compute \( u^i \) form \( i \) by repeated squaring.

2. \( \frac{1}{\sqrt{n}} \sum_{i=0}^{n-1} |i\rangle|u^i\rangle \)

   Measure the second register.

3. \( \frac{1}{\sqrt{|H_i|}} \sum_{k \in H_i} |k\rangle =: |H_i\rangle \)

   where \( H_i = \{ k \in \mathbb{Z}_n | u^k = u^i \} \).

   - \( i \in H_i \) and \( H_i = i + H = \{ i + k | k \in H \} \),
     where \( H = H_0 \).

   - the order of \( u \) is the smallest element of \( H \).
Order finding algorithm 2.

- for every \( i, k \in H \Leftrightarrow k + H_i = H_i \)

\[ \updownarrow \]

- for every \( i, \) \( \text{Shift}_k |H_i\rangle = |H_i\rangle, \)

where \( \text{Shift}_k \sum_i \alpha_i |i\rangle = \sum \alpha_i |i + k\rangle \)

- \( |H_i\rangle \) is an eigenvector with eigenvalue 1 of \( \text{Shift}_k. \)

- convenient to work with the common eigenvectors of \( \text{Shift}_k \) \( (k = 0, 1, \ldots) \)

- \( \text{Shift}_k = \text{Shift}_1^k \) are unitary transformation on \( \mathbb{C}^n, \)

have (common) orthonormal bases of eigenvectors
Order finding algorithm 3.

- The eigenvector of $\text{Shift}_1$ with eigenvalue $\omega^j$:

$$|w_j\rangle = \frac{1}{\sqrt{n}} \sum_{i=0}^{n} \omega^{-ji} |i\rangle.$$ 

- $\sum_{i=0}^{n-1} \alpha_i |i\rangle = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} \alpha_i \omega^{ij} |w_j\rangle,$

- basis transformation done by the Fourier transform:

$$\sum_{i=0}^{n-1} \alpha_i |i\rangle \mapsto \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} \alpha_i \omega^{ij} |j\rangle.$$ 

4. Do the Fourier transform, measure in the (eigen)basis $|w_j\rangle$. 
Order finding algorithm 4.

4. Do the Fourier transform, measure in the eigenbasis $|w_j\rangle$.

- If the eigenvalue of $Shift_k \ (k \in H)$ is not 1 on $w_j$ then $Prob(j) = 0$, because $|H_i\rangle$ has no components with eigenvalue not 1 under $Shift_k \ (k \in H)$

- Other $j$'s have equal probability (needs computation).

- With good probability, get $j$ that generates the group

$$\{j \in \mathbb{Z}_n | \omega^{jk} = 1 \text{ for every } k \in H\} = H^\perp = \{j \in \mathbb{Z}_n | jk = 0 \text{ for every } k \in H\}.$$

5. Then $H = j^\perp = \{k \in \mathbb{Z}_n | jk = 0\}$
Again, we assume that a multiple of the orders are known. (In view of order finding, not really restrictive assumption.)

Given $u, v$ in a group (say, $u, v \in \mathbb{Z}_N^*$) and $n \in \mathbb{Z}_{>0}$ s.t. $u^n = v^n = 0$. Find an integer $t$ such that $v = u^t$ (if exists).

Instead we will find the set

$$H = \{(k, k') \in \mathbb{Z}_n^2 | u^k v^{-k'} = 1\}.$$  

$u^t = v \iff (t, 1) \in H.$
Discrete log algorithm 1

1. \[
\frac{1}{\sqrt{n}} \sum_{i,i'=0}^{n-1} |i, i'\rangle|1\rangle
\]

2. \[
\frac{1}{\sqrt{n}} \sum_{i=0}^{n-1} |i, i'\rangle u^i v^{-i'}
\]

Measure the last register.

3. \[
\frac{1}{\sqrt{|H_{ii'}|}} \sum_{k,k' \in H_{ii'}} |k, k'\rangle =: |H_{ii'}\rangle
\]

where

\[H_{i,i'} = \{(k, k') \in \mathbb{Z}_2^n | u^k v^{-k'} = u^i v^{-i'} \}.\]

- \((i, i') \in H_{ii'}\) and \(H_{ii'} = (i, i') + H\), where \(H = H_00\).
- for every \(i, i', (k, k') \in H \iff |H_{ii'}\rangle\) is an eigenvector with eigenvalue 1 of \(\text{Shift}_{kk'}\), where

\[
\text{Shift}_{kk'} \sum_{i,i'} \alpha_{ii'} |i, i'\rangle = \sum_{i,i'} \alpha_{ii'} |i + k, i' + k'\rangle.
\]
Discrete log algorithm 2.

- \( \text{Shift}_{kk'} = \text{Shift}^k_{10} \text{Shift}^k_{01} \) are unitary transformations on \( \mathbb{C}^{n^2} \), have (common) orthonormal bases of eigenvectors;
- The common eigenvectors are

\[
|w_{jj'}\rangle = \frac{1}{n} \sum_{ii'=0}^{n} \omega^{-ji-j'i'} |i, i'\rangle.
\]

- \( \sum_{i,i'=0}^{n-1} \alpha_{i,i'} |i, i'\rangle = \frac{1}{n} \sum_{j,j'=0}^{n-1} \sum_{i,i'=0}^{n-1} \alpha_{ii'} \omega^{jj'+i'j'} |w_{jj'}\rangle, \)
  - basis transformation done by the Fourier transform in \( |i\rangle \) and than by a Fourier transform in \( |i'\rangle \)
  \[
  \sum_{i,i'=0}^{n-1} \alpha_{i,i'} |i, i'\rangle \leftrightarrow \frac{1}{n} \sum_{j,j'=0}^{n-1} \sum_{i,i'=0}^{n-1} \alpha_{ii'} \omega^{jj'+i'j'} |jj'\rangle.
  \]

4 Do the Fourier transform, measure in the eigenbasis \( |w_{jj'}\rangle \).
Discrete log algorithm 3.

- If eigenvalue of $\text{Shift}_{kk'} ((k, k') \in H)$ is not 1 on $w_{jj'}$, then $\text{Prob}((j, j')) = 0$ (easy)
- other $(j, j')$’s have equal probability (needs computation).
- with constant probability, in two steps we get $(j_1, j'_1)$ and $(j_2, j'_2)$ that generate the group
  \[
  \{(j, j') \in \mathbb{Z}_n^2 | \omega^{jk + j'k'} = 1 \text{ for every } (k, k') \in H\}
  \]
  \[= H^\perp = \{(j, j') \in \mathbb{Z}_n^2 | jk + j'k' = 0 \ \forall (k, k') \in H\}\]

5 Then $H = \{(j_1, j'_1), (j_2, j'_2)\}^\perp$
  \[
  = \{(k, k') \in \mathbb{Z}_n | j_1 k + j'_1 k' = j_2 k + j'_2 k' = 0\}.\]
1 Order finding
   - Order finding - the problem
   - Order finding algorithm

2 Discrete log
   - Discrete log - the problem
   - Discrete log - the algorithm

3 The HSP
   - Common features of order finding and discrete log
   - Generalizations
   - The HSP
   - The Graph isomorphism problem
Common features of order finding and discrete log

(and of Simon’s algorithm)

- Work in a abelian group $G$ acting as unitary transformations. 
  ($G = \{ \text{the shifts} \}$.)
- Start with the uniform superposition over $G$.
- In superposition, compute all the values of a function $f$ on $G$ in poly time.
- $f(x) = f(y)$ if $x$ and $y$ is in the same coset of a subgroup $H$.
- measuring the value gives the uniform superposition of a random coset of $H$.
- such a state is an common eigenvector of every element of $H$. 
Common features of order and discrete log 2.

- Measure in a basis consisting of common eigenvectors of $H$.
- Eigenvectors with nonzero eigenvalue under some $h \in H$ have zero probability,
- the others are equal
- Collect generators of the group ”dual” to $H$.
- Obtain $H$ by re-dualization.

Remark: Simon’s problem is in $\mathbb{Z}_2^n$. 
The problems generalize to a problem including the graph isomorphism

The method does not generalize to noncommutative groups

but generalizes to commutative groups

Why: Common eigenvectors exist in the commutative case, much weaker can be stated in the noncommutative case.

This course: What can be done in the noncommutative case.
HSP - the hidden subgroup problem

- **G** (finite) group
- **f : G → {objects}** hides the subgroup **H ≤ G**, if
  \[ f(x) = f(y) \iff xH = yH \]
  i.e., \(x\) and \(y\) are in the same left coset of \(H\).
  - In words, \(f\) is constant on the left cosets of \(H\) and takes different values on different cosets.
- \(f\) is provided by an oracle (or an efficient algorithm)
  performing \(|x⟩|0⟩ \mapsto |x⟩|f(x)⟩\)
- Task: find (generators for) \(H\).
- Examples:
  - **Order** \(G = \mathbb{Z}_n, f(k) = u^k, H = Z_{n/m},\) where \(m\) is the order of \(u\).
  - **Discrete log** \(G = \mathbb{Z}_n \times \mathbb{Z}_n, f(k, ℓ) = u^k v^{-ℓ}, H = \{(k, ℓ) = u^k = v^ℓ\}.\)
Graph automorphism

permuted graph

Γ graph on \{1, \ldots, n\}, \sigma \in S_n,
permuted graph \sigma(\Gamma), with edges:
(\sigma(i), \sigma(j)) where (i, j) edge of \Gamma.

Graph automorphism as HSP

- \( G = S_n \ f(\sigma) = \sigma(\Gamma) \).
- hidden subgroup = \( Aut(G) \)

Graph iso \leftarrow Graph auto

- \( \Gamma_1, \Gamma_2 \) connected.
- \( \Gamma_1 \cong \Gamma_2 \) iff
  \[ |Aut(\Gamma_1 \cup \Gamma_2)| = 2 \cdot |Aut(\Gamma_1)| \cdot |Aut(\Gamma_2)|. \]