1 Computational Group Theory

- Computational Questions about Groups:
  - Given a group $G$, compute $|G|$.
  - Given a group $G$, is $G$ abelian? cyclic?
  - Given a group $G$ and a subgroup $H \leq G$, is $H$ normal?
  - Given a group $G$ and a subgroup $H \leq G$ and $g \in G$, is $g \in H$?
  - etc.

- How can we be “given a group”?
  - Cayley table
  - “Black Box” that performs group operations
  - Generators and Relations
  - As a subgroup of a familiar group (e.g. $S_n$, $GL_n$)

2 Groups given as Cayley Tables or Black Boxes

- Most problems are easy to solve in time polynomial time in the size of the Cayley table ($|G|^2$).
- Can often do better with randomization.

- Randomized Algorithm to Test whether $G$ is abelian:
  1. Choose random elements $x, y \overset{R}{\leftarrow} G$.
  2. Accept if $xy \neq yx$.

- Running time $O(\log |G|)$.

- Analysis: Always accepts if $G$ abelian. If $G$ nonabelian, then:
  - $x \notin Z(G)$ with probability at least 1/2.
  - Given that $x \notin Z(G)$, the probability that $y \notin C(x) = \{z : xz = zx\}$ is at least 1/2.
  - Thus the algorithm rejects with probability at least 1/4. (Can be amplified by repetition.)

- Black-Box Groups
– Above algorithm best viewed as operating on a “black-box group”: group given as a “black box” that in one time step, can multiply two group elements, invert a group element, and test whether two group elements are equal.
– Also need to generate uniformly random group elements. Not usually part of the definition of a “black-box group” so need to give algorithms for doing this (e.g. given generators for $G$).

3 Groups given via Generators and Relations

• **Def:** For a group $G$ and a set $S \subseteq G$, we define the *subgroup generated by* $S$ to be

\[
\langle S \rangle = \{g_1g_2 \cdots g_\ell : \ell \geq 0, g_i \in S \cup S^{-1}\}
\]

= smallest subgroup of $G$ containing $S$, where $S^{-1} = \{s^{-1} : s \in S\}$. If $G = \langle S \rangle$ for a finite set $S$, we say that $G$ is *finitely generated*.

• **Examples:**
  – Finite groups?
  – Cyclic groups?
  – $\mathbb{Z}$?
  – $\mathbb{Z} \times \mathbb{Z}$?
  – $\mathbb{Q}$?

• To specify a group, it is not enough to specify the generators; we also need a set $T$ of relations between them.

• **What are the following groups?**
  – $S = \{a\}, T = \emptyset$.
  – $S = \{a, b\}, T = \{ab = ba\}$.
  – $S = \{a\}, T = \{a^n = \varepsilon\}$.
  – $S = \{a, b\}, T = \{a^n = \varepsilon, b^2 = \varepsilon, ab = ba^{-1}\}$.

• **Def:** A group is *finitely presented* if it can be described by a finite set $S$ of generators with a finite set $T$ of relations among them.

• **More formal description of generators and relations.**
  – **Def:** For a set $S$ of symbols, the *free group* on $S$ is the group $F(S)$ consisting of all strings (words) over the alphabet $S \cup S^{-1}$ with no occurrences of any substrings of the form $ss^{-1}$ or $s^{-1}s$ for $s \in S$, and with multiplication defined by cancelling inverses in the obvious way.

\[\ast \text{ e.g. in } F(\{a, b\}), (aab^{-1}a^{-1}bba^{-1})(ab^{-1}ab) = \]

2
– If \( G = \langle S \rangle \), then there is a surjective homomorphism \( \varphi : F(S) \to G \) where \( \varphi(w_1w_2\ldots w_n) = w_1 \cdot w_2 \cdots w_n \), where the latter product is in \( G \). The relations among \( S \) in \( G \) are precisely \( \{ w = \varepsilon : w \in \ker(\varphi) \} \). A presentation of \( G \) are sets \( S \subseteq G \) and \( T \subseteq F(S) \) such that \( \ker(\varphi) \) is the smallest normal subgroup of \( F(S) \) containing \( T \) (i.e. every element of \( \ker(\varphi) \) can be obtained by a finite number of multiplications by elements of \( T \cup T^{-1} \) and conjugations by elements of \( G \)).

• Unfortunately, most problems about finitely presented groups provably have no algorithms! For example:
  – Given finite \( S \) and \( T \subseteq F(S) \), is \( F(S)/\langle\langle T \rangle\rangle = \{ \varepsilon \} \)?
  – Given finite \( S, T \subseteq F(S) \), and \( w \in F(S) \), is \( w \in \langle T \rangle \)?
• Nevertheless, there are heuristics that work in some cases (Todd-Coxeter Algorithm).

4 Permutation Group Algorithms

• If \( G = \langle S \rangle \) for some given \( S \subseteq S_n \), many problems can be solved in polynomial time, namely time \( \text{poly}(n, |S|) \), even though \( G \) can be of size as large as \( n! \).

• Schreier-Sims Algorithm: from \( S \), constructs a strong generating set, which consists of:
  – A sequence \( T_1 \supset T_2 \supset \cdots \supset T_m \supset T_{m+1} = \emptyset \) of subsets of \( S_n \).
  – A base sequence \( (\beta_1, \ldots, \beta_m) \), with each \( \beta_i \in [n] \).
  such that
    – \( \langle T_i \rangle = \text{stab}_G(\beta_1) \cap \text{stab}_G(\beta_2) \cap \cdots \cap \text{stab}_G(\beta_{i-1}) \).
    – \( \langle T_{i+1} \rangle = \text{stab}_{\langle T_i \rangle}(\beta_i) \) is a proper subgroup of \( \langle T_i \rangle \).
    – \( \text{stab}_G(\beta_1) \cap \text{stab}_G(\beta_2) \cap \cdots \cap \text{stab}_G(\beta_m) = \{ \varepsilon \} \).

• Given a strong generating set, can compute \( |G| \) as follows:

\[
|G| = |\langle T_1 \rangle| = [\langle T_1 \rangle : \langle T_2 \rangle] \cdot |\langle T_2 \rangle| = [\langle T_1 \rangle : \langle T_2 \rangle] \cdot [\langle T_2 \rangle : \langle T_3 \rangle] \cdots [\langle T_m \rangle : \langle T_{m+1} \rangle] = |\text{orb}_{\langle T_1 \rangle}(\beta_1)| \cdot |\text{orb}_{\langle T_2 \rangle}(\beta_2)| \cdots |\text{orb}_{\langle T_m \rangle}(\beta_m)|
\]

• Note that \( m \leq \log_2 |G| \).

• Computing \( \text{orb}_{\langle T_i \rangle}(\beta) \):
  1. Let \( B = \{ \beta \} \).
  2. Repeat \( n \) times: Let \( B \leftarrow B \cup \{ \sigma(b) : \sigma \in T, b \in B \} \).
  3. Output \( B \).
- At the same time as computing orbits, we can compute a set $R_i$ of representatives of the cosets of $\langle T_{i+1} \rangle$ in $\langle T_i \rangle$. So that for every $b \in \text{orb}_{\langle T_i \rangle}(\beta_i)$, there is a unique $\rho \in R_i$ such that $\rho(\beta_i) = b$.

- **Claim:** Every element $\sigma \in G$ has a unique representation as a product $\rho_1 \rho_2 \cdots \rho_m$, where $\rho_i \in R_i$, and this representation can be found in polynomial time. (Thus we can test membership in $G$ by seeing if this algorithm succeeds.)

- **Algorithm:** On input $\sigma$:
  - Let $\sigma_1 = \sigma$.
  - Find unique $\rho_1 \in R_1$ such that $\rho_1(\beta_1) = \sigma_1(\beta_1)$.
  - Let $\sigma_2 = \rho_1^{-1} \sigma_1 \in \langle T_2 \rangle$.
  - Find unique $\rho_2 \in R_2$ such that $\rho_2(\beta_2) = \sigma_2(\beta_2)$.
  - Let $\sigma_3 = \rho_2^{-1} \sigma_2 \in \langle T_3 \rangle$.
  - $\vdots$
  - Find unique $\rho_m \in R_m$ such that $\rho_m(\beta_m) = \sigma_m(\beta_m)$.
  - Output $\rho_1 \cdots \rho_m$.

- **Example:** $T_1 = \{ (135), (15) \} \supset T_2 = \{ (15) \} \supset T_3 = \emptyset$, $(\beta_1, \beta_2) = (3, 1)$ is a strong generating set for $G = \langle T_1 \rangle$.
  - Compute orbits $\text{orb}_{\langle T_1 \rangle}(3)$, $\text{orb}_{\langle T_2 \rangle}(1)$, coset representatives $R_1, R_2$, and $|G|$.
  - Decide whether $(13) \in G$, $(12) \in G$. 
