Permutation Groups
Last Time

Orbit/Stabilizer algorithm:

Orbit of a point.

Transversal of transporter elements.

Generators for stabilizer.

**Today:** Use in a `divide-and-conquer` approach for permutation groups.
Some Fundamental Tasks

Groups of permutations of degree up to a few $10^6$, order easily $10^9$ (so element orbit approach is infeasible), we want to solve:

**ORDER**: find the order of a group. (Implies element membership test.)

**HOMOMORPHISM**: decompose element as generator product. (Rewriting problem, Constructive Membership.)

We want to identify the groups composition **STRUCTURE**, possibly identify composition factors.

Also subgroup centralizers, normalizers, if index is huge.
cube := Group((1,3,8,6)(2,5,7,4)(9,33,25,17)(10,34,26,18)(11,35,27,19),
> (9,11,16,14)(10,13,15,12)(1,17,41,40)(4,20,44,37)....);
<permutation group with 6 generators>
Size(cube);
43252003274489856000
DisplayCompositionSeries(cube);
G (6 gens, size 43252003274489856000)
| Z(2)
S (12 gens, size 21626001637244928000)
| A(8) ~ A(3,2) = L(4,2) ~ D(3,2) = O+(6,2)
S (9 gens, size 1072718335180800)
| Z(3)
| 7 copies in total
S (21 gens, size 490497638400)
| A(12)
S (11 gens, size 2048)
| Z(2)
| 11 copies in total
1 (0 gens, size 1)
Use Subgroups

The principal idea now is to use subgroups/cosets to factor the problem: As \(|G| = |U| \cdot [G:U]|\) this logarithmizes the problem.

Suitable subgroups: Point stabilizers \(U = \text{Stab}_G(\omega)\), index at most \(|\Omega|\).

We can iterate this process for \(U\) thus storage \(|\Omega|^{\log_2(|G|)}\).

Caveat: This works for any group and any action (matrix, automorphism, etc.) but often the problem is that \([G:\text{Stab}_G(\omega)]\) is not small.

Case in point: \(\text{GL}_n(q)\), orbit length \(q^n\).
Lemma: Any subgroup chain $G > U_1 > U_2 > \ldots > U_k = \langle 1 \rangle$ has at most $\log_2(|G|)$ steps.

Proof: If $U > V$, $U \neq V$, then $[U:V] \geq 2$, so $|U| \geq 2|V|$. Thus $|G| \geq 2^k$.

We can iterate this process for $U$ thus storage $|\Omega|^{\log_2(|G|)}$.

Caveat: This works for any group and any action (matrix, automorphism, etc.) but often the problem is that $[G:\text{Stab}_G(\omega)]$ is not small.

Case in point: $\text{GL}_n(q)$, orbit length $q^n$. 
Stabilizer Chains

Let $G \leq S_\Omega$. A list of points $B=(\beta_1,\ldots,\beta_m)$, $\beta_i \in \Omega$ is called a base, if the identity is the only element $g \in G$ such that $\beta_i^g = \beta_i$ for all $i$.

The associated Stabilizer Chain is the sequence

$$G = G^{(0)} > G^{(1)} > \ldots > G^{(m)} = \langle 1 \rangle$$

defined by $G^{(0)} := G$, $G^{(i)} := \text{Stab}_{G^{(i-1)}}(\beta_i)$. (Base property guarantees that $G^{(m)} = \langle 1 \rangle$.)

Note that every $g \in G$ is defined uniquely by base images $\beta_1^g,\ldots,\beta_m^g$. (If $g,h$ have same images, then $g/h$ fixes base.)
Base Length

The base length $m$ often is short ($m \leq \log_2(|G|)$). In practice often $m < 10$.

We say that $G$ is short-base if $\log |G| \leq \log^c |\Omega|$ for some $c$. (Important to make polynomial in input size!)

Bounds on base length have been studied in theory. If there is no short base the groups must be essentially alternating $A_n$ and relatives/products.

Same concept also possible for other kinds of groups, or different actions, but then no good orbit length/base length estimates.
Data structure

We will store for a stabilizer chain:

- The base points \((\beta_1, \ldots, \beta_m)\).
- Generators for all stabilizers \(G^{(i)}\). (Union of all generators is \textit{strong generating set}, as it permits reconstruction of the \(G^{(i)}\).) Data structure thus is often called \textbf{Base} and \textbf{Strong Generating Set}.
- The orbit of \(\beta_i\) under \(G^{(i-1)}\) and an associated transversal for \(G^{(i)}\) in \(G^{(i-1)}\) (possibly as \textit{Schreier tree}).

Storage cost thus is \(\mathcal{O}(m \cdot |\Omega|)\)
Consequences

- Group order: \( G = [G^{(0)}:G^{(1)}] \ldots [G^{(m-1)}:G^{(m)}] \) and thus \( G = \prod_i |\beta_i^{G(i-1)}| \).

- Membership test in \( G \) for \( x \in S\Omega \):
  1. Is \( \omega = \beta_1^x \in \beta_1^G \)? If not, terminate.
  2. If so, find transversal element \( t \in G^{(0)} \) such that \( \beta_1^t = \beta_1^x \).
  3. Recursively: Is \( x/t \) (stabilizing \( \beta_1 \)) in \( G^{(1)} \).
    (Respectively: test \( x/y = (\ ) \) in last step.)
More Consequences

Bijection $g \in G \Leftrightarrow$ base image ($\beta_{1g}, \beta_{2g}, \ldots$).

- Enumerate $G$, equal distribution random elements.
- Write $g \in G$ as product in transversal elts.
- Write $g \in G$ as product in strong generators.
- If strong gens. are words in group gens.: Write $g \in G$ in generators of $G$. (Caveat: Long words)
- Chosen base: Find stabilizers, transporter elements, for point tuples.
More Consequences

Bijection $g \in G \iff$ base image ($\beta_1, \beta_2, \ldots$).

- Enumerate $G$, equal distribution random elements.
- Write $g \in G$.
- Compute Bijection $G \leftrightarrow \{1, 2, \ldots, |G|\}$
- Write $g \in G$ as product in strong generators.
- If strong gens. are words in group gens.: Write $g \in G$ in generators of $G$. (Caveat: Long words)
- Chosen base: Find stabilizers, transporter elements, for point tuples.
Generic Decomposition

remember:

cube:=Group(top,left,front);
map:=EpimorphismFromFreeGroup(cube:names:=["T","L","F"]);
Factorization(cube,(1,22,8)(2,17,14)(3,7,6,20,9,23)(5,12,11)(15,21,18));

returned $L^{-1}T^{-1}FL^{-1}L^{-1}F^2L^{-1}T$, but at high memory cost.

PreImagesRepresentative(map,(1,22,8)(2,17,14)(3,7,6,20,9,23)(5,12,11)(15,21,18));

returns $T^{-1}F^2L^{-1}TL^{-1}T^{-1}FT^{-1}F^{-1}T^{-3}L^{-1}TLTL^{-1}TF^{-1}LFT^{-1}LT^{-1}$

Faster, less memory, but longer word.

Heuristics on trying short Schreier generators gets 3x3x3 cube word length to ~100 (naively: Millions)
Schreier-Sims algorithm

Sims' (~1970) idea is to use a membership test in a partially completed stabilizer chain to reduce on the number of Schreier generators.

Basic structure is a partial stabilizer, i.e. a subgroup $U \leq G^{(i-1)}$ given by generators and a base-point orbit $\beta^U$ with transversal elements (products of the generators of $U$).

The basic operation now is to pass an element $x \in G^{(i-1)}$ to this structure and to consider the base point image $\omega = \beta^x$. 
Base point image $\omega = \beta^x$

- If $\omega \epsilon \beta^U$, transversal element $t \epsilon U$ such that $\beta^t = \omega$. Pass $y = x/t$ to the next lower partial stabilizer $\leq G(i)$. If $\omega \notin \beta^U$, add $x$ to the generating set for $U$ and extend the orbit of $\beta$. All new Schreier generators $y \epsilon \text{Stab}_U(\beta)$ are passed to next partial stab. $\leq G(i)$. If no lower stabilizer was known, test whether the $y$ was the identity. If so just return. (Successful membership test.) Otherwise start new stabilizer for generator $y$ and the next base point. (Pick a point moved by $y$).
Homomorphisms

Embed permutation group $G$ into direct product $D = G \times H$. A homomorphism $\varphi : G \rightarrow H$ can be represented as $U \leq D$ via

$$U = \{(g, h) \in G \times H \mid g \varphi = h\}$$

Build a stabilizer chain for $U$ using only the $G$-part.

Then decomposing $g \in G$ using this chain produces an $H$-part that is $g^{(\varphi^{-1})}$. Use this to evaluate arbitrary homomorphisms.
Kernels, Relators

Vice versa, let \( \varphi : H \rightarrow G \) and 
\[
U = \{(g,h) \in G \times H \mid h \varphi = g\}.
\]

Form a stabilizer chain from generators of \( U \), using the \( G \)-part.
The elements sifting through this chain (trivial \( g \)-part) are generators for \( \ker \varphi \).

If \( H \) is a free group, this yields relators (later) of a presentation for \( G \).
Backtrack

Algorithmic technique that stems from the idea of solving a maze (labyrinth): *Keep the right hand on the wall.*

This method enters paths and returns "back" if they are not successful.

Analog method for tree transversal — depth first visit of leaves.
Permutation Backtrack

A stabilizer chain lets us consider the elements of $G$ as leaves on a tree, branches corresponding to base point images.

Traverse the tree (depth first) by enumerating all possible base images. Find group elements with particular desired property.

Exponential run time but good in practice, by clever pruning.

E.g.: Centralizer, Normalizer, Set Stab., Intersection, conj.elts., ...
Search Tree Pruning

It is crucial to reduce the search space down from $|G|$ to a manageable size. Tools:

**Algebraic structure:** Solution set forms a subgroup (if stabilizer) or double coset (if transporter). E.g., all elements mapping $\omega$ to $\delta$ lie in $\text{Stab}_G(\omega) \cdot g \cdot \text{Stab}_G(\delta)$ where $\omega^g = \delta$.

The closure properties of the structure mean that the existence of some elements implies existence of others. For simplicity, assume that we are aiming to find $S = \text{Stab}_G(\omega)$. 
Double Coset Pruning

Assume we have found (or were given) some elements of $S$, generating subgroup $U$. (Hard part is to prove there are no further ones.)

If $g \in G$, then either all or no elements of $UgU$ will be in $S$. Sufficient to test one.

**Criterion:** Only test $g$ if it is minimal in $UgU$. (lexicographically as lists of base images.)

Minimal in $UgU$ is hard. Instead use minimal in $Ug$ and in $gU$ (necessary, not sufficient). Restrict choice of possible base images.
Problem-specific Pruning

The real power of backtrack comes with pruning methods that are specific to the problem to be solved. For example:

- An element centralizing a permutation must map cycles to cycles of the same length. Images of the first cycle point thus are limited. Once the image $\omega^g$ of a first cycle point is chosen, the images of all other points in the cycle are given.

- An element normalizing a subgroup $U$ must preserve the orbits of $U$. When also fixing the point $\omega$, one must preserve the orbits of $\text{Stab}_U(\omega)$ (these are called orbitals).
Base Change

For efficiency, it is helpful to use a base that causes problem-specific prunings to apply early.

E.g. when centralizing an element, choose the first base point in a cycle of longest length (as the choice of one point image determines all others).

There used to be algorithms that performed a base change, i.e. computed a new stabilizer chain from an old one but with different base.

Modern, randomized, Schreier-Sims algorithms are so fast that is is usually easiest to just compute a new chain for the desired base.
Partition Backtrack (MCKAY, LEON, THEISSEN,...) is a convenient way to process the different kinds of pruning.

The algorithm maintains a partition (list of points) of $\Omega$, indicating possible images of the base points. Tree root=$\Omega$, leaves=1 point cells.

Selection of base images and pruning conditions are partition refinements, done by intersecting with particular partitions, such as (img, rest) or orbits of a subgroup.
Quandry

Schreier-Sims is deterministic Algorithm. Polynomial (in the degree \( n=|\Omega| \)) runtime, but larger exponent (naively \( \Theta(n^7) \), \( \Theta(n^3) \) if Schreier tree is used cleverly).

The cause is the processing of all (mostly redundant) Schreier generators.

In practice this is not feasible if \( n \) is not small (\( >1000 \)). For short base (\( \log|G| \leq \log^c n \)) we would like nearly linear time \( \Theta(n \log^c n) \), best possible
Wrong Results are Cheap

Use only *some* generators (random subset, better: random subproducts, see orbit slides). Potentially wrong data structure. But:

▷ Error results in chain that claims to be too small - can detect if group order is known. (Random-Schreier-Sims)

▷ Error analysis: A random element of $G$ fails sifting in wrong chain with probability $1/2$ - guarantee arbitrary small error probability.

▷ Verify correctness, by showing the group cannot be larger:
  ➡ Combinatorial Verification (Sims, see SERESS' book)
  ➡ Presentation from stabilizer chain. Verify that group fulfills it. If too small, some relators fail to be. (Todd-Coxeter-Schreier-Sims; or Recognition, see later)
Problems:

1. How large is the action of 3x3x3 Rubik’s cube on the 8 corners? On the 12 edges? Why is the order not the product?

2. If we put pictures on the sides, rotations of the middle faces count. How much larger does the group get?

3. Can you make a conjecture about the structure (order, composition factors) of an $n \times n \times n$ cube? (Hint: can you give different thicknesses to the slices?)
Finitely Presented Groups
In Linear Algebra

Two ways of describing a subspace:

• By a basis (generators)

• As nullspace of a linear transformation.

Does this second model also work for groups?
Presentations

First textbook example of a nonabelian group:

\[ D_8 = \langle r, s \mid r^4 = s^2 = 1, rs = sr^{-1} \rangle \]

- Elements have normal form \( s^a r^b \), \( 0 \leq a \leq 1, 0 \leq b \leq 3 \).

- So group has at most 8 elements.

- And indeed there are 8 different ones (show that elements have different actions).

What does this notation mean?

Can we generalize?
Presentations

First textbook example of a nonabelian group:

\[ D_8 = \langle r, s \mid r^4 = s^2 = 1, rs = sr^{-1} \rangle \]

- Elements have normal form \( s^a r^b \), \( 0 \leq a \leq 1 \), \( 0 \leq b \leq 3 \).

There are 8 different ones (show that different actions).

What does this mean?
Presentations

First textbook example of a non-abelian group:

\[ D_8 = \langle r, s \mid r^4 = s^2 = 1, rs = sr^{-1} \rangle \]

• Elements have normal form \( s^a r^b \), \( 0 \leq a \leq 1, 0 \leq b \leq 3 \).

First Spiegelung (ger.) = reflection

There are 8 different ones (show that there are 8 different actions).

What does this mean?
Presentations

First textbook example of a nonabelian group:

\[ D_8 = \langle r, s \mid r^4 = s^2 = 1, rs = sr^{-1} \rangle \]

- Elements have normal form \( s^a r^b \), \( 0 \leq a \leq 1 \), \( 0 \leq b \leq 3 \).

There are 8 different ones (show that different actions).

What does \( r \) mean?

rotation

1 4

S

2

3

Spiegelung (ger.)
= reflection
Presentations

First textbook example of a nonabelian group:

\[ D_8 = \langle r, s \mid r^4 = s^2 = 1, rs = sr^{-1} \rangle \]

• Elements have normal form \( s^a r^b, 0 \leq a \leq 1, 0 \leq b \leq 3 \).

• So group has at most 8 elements.

• And indeed there are 8 different ones (show that elements have different actions).

What does this notation mean?

Can we generalize?
Words And Free Groups

Take an alphabet $A$ of symbols $a,b,c,\ldots$

A *word* in $A$ is a sequence of symbols from $A$: $acbabc$.

The *free group* on $A$ is the set of all words in $A$ and $A^{-1}$ (formal inverses): $ac^{-1}ba^{-1}bc$.

Identity: empty word

Product: Concatenation and cancellation of inverses.

*Exercise*: This is associative. (Cancellation forces cases)
Finitely Presented Group

Let $F$ be a free group on finite alphabet $A$.

Let $R$ (*Relators / Relations*) be a finite subset of $F$.

(Interpret $a=b$ as $a/b$.)

Then $\langle A \mid R \rangle = F/N$, where $N$ is the smallest normal subgroup of $F$ containing $R$, is a *finitely presented group*.

Some people distinguish the groups (quotients of $F$) and *presentations* (formal string objects) to describe modifications.
Elements

Elements of group \( \langle A \mid R \rangle \) are represented by words from \( F \).

Equivalence: Finite number of “applications” of \( R \). This is hard to test, as there is no length bound.

But: Nice, Compact, Natural, way of representing groups.
Natural

The fundamental group of a topological space is set of all closed paths from a chosen point up to homotopy.
Example: Torus

\langle a, b \mid ab a^{-1} b^{-1} = 1 \rangle
Generators, Relations

1. Take an (equivalent) simplicial complex. (Edges, Triangles, …)
Generators, Relations 2

2. Form a spanning tree. Generators are non-tree edges.
3. For a triangle in the complex, relator $xyz^{-1}$ if loop around triangle. (tree edges are 1).

- $a11^{-1}$
- $b$
- $c$
- $d$
- $cf^{-1}$
- $de$

but not $g$
3. For a triangle in the complex, relator $xyz^{-1}$ if loop around triangle. (tree edges are 1).

**Theorem:** This presents the fundamental group.
DEHN's Problems (1911)

Given a finitely presented group, can we test algorithmically whether:

1. A word represents the identity
2. Two words are conjugate elements
3. Test for Group Isomorphism
Given a finite group, it is undecidable to determine if the group is abelian.

1. Algorithmically present the identity element.
2. Test for group isomorphism.

William Boone, 1920-1983
Piotr Novikov, 1901-1975

Answer: Boone, Novikov (1957) No.

Reduce to Halteproblem for Turing machines — there is no universal Turing machine. All will be heuristics.