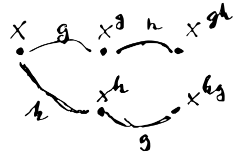
Transversals:

we know that $x^G \leftrightarrow \mathrm{Stab}_G(x) \backslash G$.

can we not only find the orbit of x, but also representatives for cosets?



$\ldots$

$[x, \; x^g, \; x^h, \; x^{gh}, \; x^{hg}, \; \ldots \;]$

$[1, \; g, \; h, \; gh, \; hg, \; \ldots \;]$

Defn:

<u>Transversal</u> is the list of coset representatives associated to orbit x^G.

Notation:

- We will blend the notions of an orbit and a Transversal.
- If T is a transversal for x^G, then $T[y] = g$ s.t. $x^g = y$.

ALGORITHM: ORBIT-TRANSVERSAL

<u>INPUT:</u>
- S - set of generators of G
- x - point to act on

<u>OUTPUT:</u>
- Δ - orbit x^G
- T - associated transversal

```
begin
   Δ = [x]
   T = [e]   <- group identity
   for δ ∈ Δ
      for s ∈ S
         γ = δ^s
         if γ ∉ Δ
            push γ to Δ
            T[γ] = T[δ]·s      <- group element which sends x to δ.
         end
      end
   end
   return Δ, T
end
```

Problem with this algorithm: storage.

1 permutation of length 2^{10} is 8 KiB

10^6 such permutations $\sim$ 10 GiB

1 perm of length 2^{20} is $1024 \cdot 8$ KiB $= 8$ MiB

512 of such is already 8 GiB

Solution: Only store the <u>minimal information required</u> to <u>reconstruct</u> elements from the transversal!

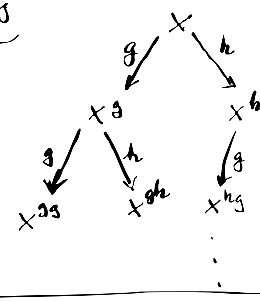
Definition:

Let $\Delta = x^G$ (as ordered set).

We say that $V = (v_1, \dots, v_k) \in (S \cup \{e\})^k$ <u>Schreier vector</u> for Δ iff

1) $k = |\Delta|$ $\qquad\qquad \Delta = [x, x^g, x^h, x^{gh}, \dots]$
2) $v_1 = e$ $\qquad\qquad V = [e, g, h, h, \dots]$
3) whenever $y \in \Delta$ and $\underbrace{v[y] = s}$ then $y^{s^{-1}}$ occurs in Δ <u>before</u> y.

the entry of v associated to y.



INPUT:
- S – a generating set for $G = \langle S \rangle$
- x – a point $\in \Omega$

OUTPUT:
- Δ – x^G, the orbit of x
- v – Schreier vector / tree

```
Δ = [x]
v = [e]
for δ ∈ Δ
    for s ∈ S
        γ = δ^s
        if γ ∉ Δ
            push γ to Δ
            push s to v     end end end return Δ, v.
```

ALGORITHM: RECONSTRUCT REPRESENTATIVE

INPUT: Δ - orbit of x
v - associated Schreier vector/tree
y - point in Δ

```
begin
    γ = y        // current point
    g = e        // coset representative
    while γ ≠ x
        s = v[γ]
        g = s·g
        γ = γ^(s^-1)
    end
    return g
end
```

Generating set for the stabilizer

Let $\mathcal{G} = \langle S \rangle = \langle s_1, \dots, s_k \rangle$ and let
$H < \mathcal{G}$ be of finite index with
coset representatives $\{r_1, \dots, r_n\}$ (we assume that $r_1 = e$).

for $g \in \mathcal{G}$ there exists a unique coset s.t.
$g \in H \cdot r_j$. we will write $\bar{g}$ for r_j.

Let $T := \{ r_i s_j (\overline{r_i s_j})^{-1} \}_{\substack{i=1,\dots,n \\ j=1,\dots,k}}$

Lemma: $H = \langle T \rangle$.

T is called the set of Schreier generators.

1) $r_i s_j (\overline{r_i s_j})^{-1} \in H$ ✓

2) let $x \in H$, $x = g_1 \cdots g_m$, $g_i \in S$.

$$x = g_1 \cdots g_m$$
$$= 1 \cdot g_1 \cdots g_m$$
$$= \underbrace{1 \cdot g_1 ((\overline{1 \cdot g_1})^{-1}}_{\in T} \cdot \underbrace{(\overline{1 \cdot g_1})}) g_2 \cdots g_m$$
$$= t_1 \cdot \overline{g_1} \cdot g_2 \cdots g_m =$$
$$= t_1 \cdot \underbrace{\overline{g_1}\, g_2 (\overline{\overline{g_1} \cdot g_2})^{-1}}_{t_2 \in T} \cdot (\overline{\overline{g_1} \cdot g_2}) \cdot g_3 \cdots g_m$$
$$= t_1 \cdot t_2 \cdot \underbrace{(\overline{\overline{g_1} \cdot g_2}) g_3 \cdot (\overline{\overline{\overline{g_1} \cdot g_2} \cdot g_3})^{-1}}_{t_3} (\quad) \cdot g_4 \cdots g_m$$

$$= t_1 \cdot t_2 \cdot t_3 \cdots t_{m-1} \underbrace{\overline{\overline{\overline{g_1} g_2} \cdots g_{m-1}}\, g_m}_{z = ???}$$

Claim: $\overline{z} = e$, i.e. $z \in H \Rightarrow z (\overline{z})^{-1} \in T$.

Application:

$H = \mathrm{Stab}_{\mathcal{G}}(x) = \{ g \in \mathcal{G} : x^g = x \}$

$\Delta \leftarrow$ orbit of x $\longleftrightarrow$ cosets of $H \backslash \mathcal{G}$.

transversal

If $\underbrace{T[x^g]}_{=g} \longleftrightarrow Hg$ ← coset

ALGORITHM: Orbit / stabilizer

INPUT:
- S – set of generators for $\mathcal{G} = \langle S \rangle$
- x – a point in Ω

OUTPUT:
- Δ – the orbit of x
- T – transversal for Δ (Schreier or not)
- $\mathcal{U}$ – set of Schreier generators

```
begin
    Δ = [x]
    T = [e]
    U = [e]
    for δ ∈ Δ
        for s ∈ S
            γ = δ^s
            if γ ∉ Δ
                push γ to Δ
                push s to T
            else
                push T[δ]·s·T[γ]^-1 to U
            end
        end
    end
    return Δ, T, U
end
```

Performance: (problem)

- $\mathcal{U}$ will have $\sim k \cdot n = |S| \cdot |\Delta|$ elements
  that's plenty, but sometimes all are needed
- remove duplicates and identify
- instead of collecting all those generators simply form a group $H = \langle \mathcal{U} \rangle$, and check if the new one already belongs to H.

Further applications:

- Normal closure of $H < \mathcal{G}$ i.e. (also known as the normaliser of H in $\mathcal{G}$).

$$\mathcal{N}_{\mathcal{G}}(H) = \bigcap \{ \mathcal{N} \triangleleft \mathcal{G} : H < \mathcal{N} \}$$

Tips: if $H = \langle \mathcal{U} \rangle$
start the orbit algorithm with $\Delta = \mathcal{U}$
under the action $g^h = h^{-1} g h$.

Proof: 1) termination

2) Suppose that $\mathcal{N} = \langle \Delta \rangle$

$H < \mathcal{N}$ since $\mathcal{U} \subset \Delta$.

3) We have $\mathcal{N} < \mathcal{N}_{\mathcal{G}}(H)$ since every elt of Δ can be written as conjugation $g u g^{-1}$ of $u \in \mathcal{U}$.

4) if $x \in \mathcal{N}_{\mathcal{G}}(H)$ it can be written as a product of conjugates of gens of H.

Finally: Pseudorandom elements

To find truly random elts in G one would need to

1) access all elts of G as a list (at arbitrary locations)

2) generate random number from $1:n$

and then pick the corresponding elt.

1) is infeasible (too many of them)

2) is impossible (no such hardware exist).

ALGORITHM: Pseudo-random

INPUT:
- X – a list of "sufficiently" random elts from G
- s – the seed

OUTPUT:
- X – a list of ______ . ______
- g – a random element

```
begin
    i, j – two distinct integers from 1:|X| chosen
                                                at random
    sgn = rand((-1, 1))
    if rand(Bool)
        X[i] = X[i] · X[j]^sgn
        g = s · X[i]
    else
        X[i] = X[j]^sgn · X[i]
        g = X[i] · s
    end
    return X, g
end
```

ALGORITHM: initialize

INPUT: • S - a set of generators for $G = \langle S \rangle$

OUTPUT: • X - a list of "sufficiently random" elements of G
• a - a pseudorandom group element

```
begin
    X = concatenate S with itself as long as |X| < 11
    a = e        // the accumulator //        // 11 is heuristic //
    for _ in 1:50      // also a heuristic
        X, a = pseudorandom(X, a)
    end
    return X, a
end
```