# Computing with finitely presented groups

Sarah Rees

University of Newcastle

ICERM, 14-18 June 2021

# Abstract


I'll talk about computation with finitely presented groups. In particular:

(1) an introduction to the basic concepts and techniques, the use of geometry via the Cayley graph, the fact that some questions are not decidable in general, other calculations are not practical,
(2) calculation with abelian and polyabelian groups, construction of quotients of that type,
(3) techniques associated with coset enumeration and subgroup presentations (incl. Todd-Coxeter, Reidemeister-Schreier),
(4) techniques associated with rewriting, including the Knuth-Bendix process, and computation and use of automatic structures,
(5) testing for hyperbolicity, conjugacy problem in hyperbolic groups.

I'll cover 1–3 in lecture 1, 4,5 in lecture 2.

# Contents: Lecture 1

# Word, relations and group presentations

Let $X = \{x_1, x_2, \ldots\}$, $X^{-1} = \{x_1^{-1}, x_2^{-1}, \ldots\}$, $X^{\pm} := X \cup X^{-1}$. A **word** $w$ over $X$, of length $n$, is a string $a_{i_1} \cdots a_{i_n}$ of symbols from $X^{\pm}$.
We delete all subwords $x_i x_i^{-1}$ or $x_i^{-1} x_i$ from $w$ to form its **free reduction**, can abbreviate this as a product of powers $a_{j_1}^{k_1} \cdots a_{j_m}^{k_m}$ ; for words $w, w'$ we write $w \sim w'$ if $w, w'$ have the same free reduction.

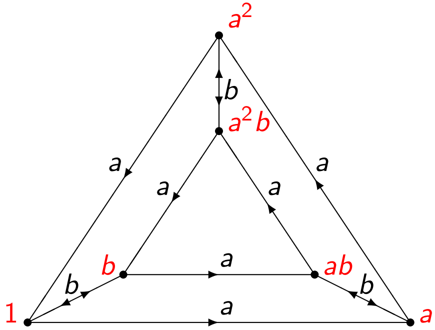
We define the **free group** $\mathbb{F}(X)$ on the set of $\sim$-equiv. classes of words:
- multiplication is defined by concatenation,
- the empty word $\epsilon$ represents the identity element,
- the word $w^{-1} := a_{i_k}^{-1} \cdots a_{i_1}^{-1}$ represents the inverse of $w$ above.

Given a set $R$ of eqns between words over $X$, we extend $\sim$ to equiv. rel. $\simeq$ by additionally relating $w, w'$ if $w = w_1 u w_2$, $w' = w_1 v w_2$, where either '$u = v$' or '$u^{-1} = v^{-1}$' is in $R$. We call '$u = v$' a **relation**, $uv^{-1}$ a **relator**. Then we define the group $G = \langle X \mid R \rangle$, on the set of $\simeq$-equiv. classes, just as above. The **presentation** $(X, R)$ is called finite (and $G$ `fp`) if $X, R$ are both finite.

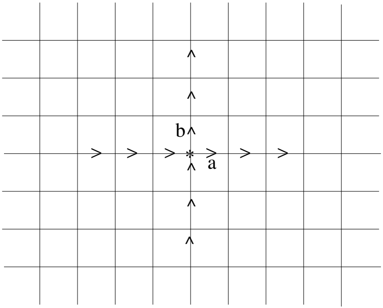
# The Cayley graph $\mathcal{G}(G, X)$ of a group $G$

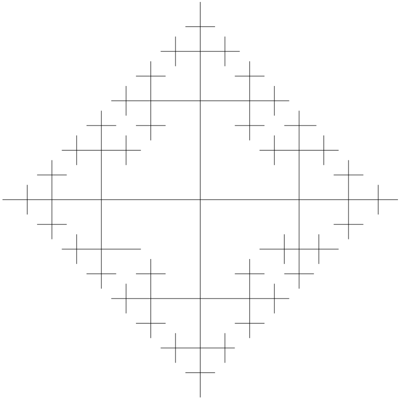
The vertices of the graph correspond to the group elements. And for each generator $x$ there's a (directed) edge from $g$ to $gx$. When a product represents an element $g$, it labels a path in the graph from vertex 1 to vertex $g$, so when a product $w$ represents 1, it labels a loop from 1 to 1.

**E.g.** $\mathcal{S}_3 = \langle a, b \mid a^3 = b^2 = 1, ba = a^3 b \rangle$



# Cayley graphs for $\mathbb{Z}^2$ and the free group $F_2$



# Some problems are insoluble, others merely hard

In 1908, Max Dehn defined his three famous decision problems for $\texttt{fg}$ groups $G = \langle X \mid R \rangle$ ($|X| < \infty$).

- The **word problem** for $G$ ($\texttt{WP}(G)$) asks whether or not a given input word represents the identity; $\texttt{WP}(G)$ is soluble if $\exists$ a terminating algorithm that can decide on any input word $w$ whether $w =_G 1$.
- The **conjugacy problem** for $G$ ($\texttt{CP}(G)$) asks whether or not two given input words $u_1, u_2$ are conjugate in $G$ (i.e. $\exists g \in G$, $gu_1 =_G u_2 g$).
- The **isomorphism problem** for a class of groups asks whether or not two given groups within the class are isomorphic.

Of the three problems, $\texttt{WP}(G)$ is the easiest. But (Novikov, Boone 1950s), $\texttt{fp}$ groups $G$ exist for which $\texttt{WP}(G)$ cannot be solved. Hence all three problem are insoluble in general.

So there are some computational problems for which we cannot hope to find solutions in all $\texttt{fp}$ groups. There are others that are theoretically solvable, but constraints of time and space limit what we can achieve.

# Abelian groups are easy

If we know that a group $G$ is abelian, calculation with it is easy. Then the group has a decomposition as a direct product of cyclic groups,

$$\langle x_1 \rangle \times \langle x_2 \rangle \times \cdots \times \langle x_r \rangle, \quad |x_i| = m_i,\ m_1 \leq m_2 \cdots m_r \leq \infty.$$

We can derive that decomposition from any presentation, using linear algebra (we'll come back to that), express every element in its normal form $x_1^{i_1} \cdots x_r^{i_r}$.

Using the normal form,

- multiplication and inversion are easy,
- orders of elements are visible,
- calculation is efficient,
- solution of word and conjugacy problems are easy (and so is isomorphism)

# Computation with polycyclic groups is not much harder . . .

. . . once we have found power conjugate/power commutator presentations, and corresponding normal forms.

A **power-commutator** presentation for a group is a presentation over generators $x_1, \ldots, x_r$ in which every relation expresses either a power of a generator $x_i$ or a commutator of two generators $x_j, x_i$ or their inverses as a word in lower numbered generators. **Power-conjugate** presentations are defined similarly, with commutators of two generators (or their inverses) replaced by conjugates. Clearly a presentation of either of these two types can easily be transformed into one of the other two types.

Any `fg` polycyclic group possesses such a presentation.

Where a group is given in this form, there is a natural normal form, consisting of elements of the form $x_1^{i_1} \cdots x_r^{i_r}$.

Hence, as in abelian groups, computation within polycyclic groups is relatively straightforward.

# Quotients of finitely presented groups

Given a finite presentation $\langle X \mid R \rangle$ for a group $G$, we can compute (information about) various types of quotients of $G$, particularly quotients that are abelian, poly-abelian, or of $p$-power order, and maximal quotients of those type (with specified parameters), as well as finite permutation groups.

We might use information about such quotients to investigate the structure of $G$, its finiteness or otherwise, its isomorphism or otherwise with another finitely presented group.

We'll discuss just abelian and polyabelian quotients for now.

# Computing the largest abelian quotient

Suppose that $G = \langle X \mid R \rangle$ is $\mathtt{fp}$, with $X = \{x_1, \ldots, x_n\}$, and $|R| = m$. The largest abelian quotient $G_{\mathrm{ab}} = G/[G, G]$ of $G$ has presentation

$$\langle X \mid R \cup \{[x, y] : x, y \in X\} \rangle.$$

For $w = 1$ in $R$, let $e_w$ be the vector of exponents of the elements of $X$ within the word $w$. Let $E_R$ be the $m \times n$ matrix whose rows are the $e_w$.

Using additive notation, $G_{\mathrm{ab}} \cong$ the abelian gp on $X$ subject to eqns $E_R \mathbf{x} = \mathbf{0}$, where $\mathbf{x} = (x_1, \ldots, x_n)^T$. Using linear alg. we change gen set to $Y = \{y_1, \ldots, y_n\}$, eqns to $D\mathbf{y} = \mathbf{0}$, $D$ diagonal, diag. entries $d_1, \ldots, d_m \geq 0$, $d_i \mid d_{i+1}$, those $\neq 1$ the **abelian invariants** for $G_{\mathrm{ab}}$. Then

$$Gab \cong \langle y_1 \rangle \oplus \cdots \oplus \langle y_n \rangle \cong \mathbb{Z}/d_1\mathbb{Z} \oplus \cdots \oplus \mathbb{Z}/d_m\mathbb{Z} \oplus \mathbb{Z}^{n-m}$$

We apply elementary row and column operations

$$r_i \to r_i + \lambda r_j,\ c_i \to c_i + \mu c_j,\ r_i \leftrightarrow r_j,\ c_i \leftrightarrow c_j, \quad \lambda, \mu \in \mathbb{Z}$$

to transform $E_R$ into $D$, its **Smith Normal Form**.

# Computing the largest abelian quotient: example

$$G = \langle x_1, x_2, x_3, x_4 \mid x_1(x_2x_4^{-1})^3,\ (x_2x_3)^3,\ x_4^3x_1^{-1}(x_2^{-1}x_3)^6$$

$$E_R = \begin{pmatrix} 1 & 3 & 0 & -3 \\ 0 & 3 & 3 & 0 \\ -1 & -6 & 6 & 3 \end{pmatrix} \xrightarrow[r_3 \to r_3 + r_1]{} \begin{pmatrix} 1 & 3 & 0 & -3 \\ 0 & 3 & 3 & 0 \\ 0 & -3 & 6 & 0 \end{pmatrix}$$

$$\xrightarrow[r_3 \to r_3 + r_2]{} \begin{pmatrix} 1 & 3 & 0 & -3 \\ 0 & 3 & 3 & 0 \\ 0 & 0 & 9 & 0 \end{pmatrix} \xrightarrow[c_4 \to c_4 + 3c_1]{c_2 \to c_2 - 3c_1} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 3 & 0 \\ 0 & 0 & 9 & 0 \end{pmatrix}$$

$$\xrightarrow[c_3 \to c_3 - c_2]{} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 9 & 0 \end{pmatrix}$$

We've found a decomposition of $G_{\text{ab}}$ as a direct sum/product of cyclic groups of orders $1, 3, 9, \infty$. Examining the column ops, we see that the new gens are the images in $G/[G, G]$ of $x_1x_2^3x_4^{-3}, x_2x_3, x_3, x_4$.

# Exponent-$p$ abelian and $p$-quotients

If we add the equations $px_i = 0$ to our system of equations, we can use linear algebra to find the largest **exponent-$p$ abelian quotient** of $G$.

Then we can iterate, move down the **lower exponent-$p$ central series**:

$$G = P_0(G) \rhd \cdots P_{i-1}(G) \rhd P_i(G) \rhd \cdots$$

where $P_i(G) = [P_{i-1}(G), G]P_{i-1}(G)^p$ for $i \geq 1$, and hence compute finite $p$-quotients $G/P_i(G)$ of increasing (exponent-$p$) nilpotency classes.

A **power-conjugate** presentation is computed for each quotient, over a generating set $a_1, \ldots, a_n$ whose relations specify, for each $a_i$, words in $a_1, \ldots, a_{i-1}$ that are equal either to some power of $a_i$ or to its conjugate by some $a_j$ with $j < i$.

The original **$p$-quotient algorithm** was developed by I.D.Macdonald (1974). The current version was introduced by Newman (1976), with contributions from Havas, O'Brien, Vaughan-Lee. Implementations are available in GAP and MAGMA.

# Other polyabelian quotients

The same basic idea (building down a series with abelian quotients, using linear algebra and other techniques) constructs

- nilpotent quotients (Nickel 95, Newman),
- polycyclic quotients (Baumslag 81, Lo 98, Sims),
- solvable quotients (Leedham-Green 84, Plesken 87, Brückner, Niemeyer 93)

again finding power-conjugate presentations.

These are resource-expensive computations, needing sophisticated optimisation techniques.

# Finite quotients of a group

It is well known that, for any group $G$, there is a bijection between its set of finite index subgroups $H$, and its set of **transitive actions** of $G$ on finite sets, represented by maps $\varphi : G \to \mathcal{S}_n : n \in \mathbb{N}$.

Acting on the right ($\omega \mapsto^g \omega g$),

$H \quad \leftrightarrow \quad$ right coset action on $\{Hy : y \in G\} =: H\backslash G$
defined by $Hy \mapsto^g Hyg$.

We have

$$\begin{array}{rcl} H & = & \varphi^{-1}(\mathrm{stab}_{\mathcal{S}_n}(1)) \\ \bigcap_{y \in G} H^y & = & \ker(\varphi) \\ H = K^y,\ y \in G & \Longleftrightarrow & \text{actions on } H\backslash G \text{ and } K\backslash G \text{ are equivalent.} \end{array}$$

So the conjugacy classes of finite index subgroups correspond to the equivalence classes of actions on finite sets.

# Coset tables

We can describe the action of $G$ on $H \backslash G$ , $|G : H| < \infty$ using a coset table. For example:

$$\begin{aligned} G &= \langle c, d \mid c^2 = 1 = d^3 = (cd)^7 = [c, d]^4 \rangle \\ &\cong PSL_2(7) \cong \langle (1,2)(4,5), (2,3,4)(5,6,7) \rangle \leq \mathcal{S}_7 \\ H &= \langle d, cdcd^{-1}c \rangle, \ |G : H| = 7. \end{aligned}$$

| Coset no. | $c$ | $c^{-1}$ | $d$ | $d^{-1}$ |
|---|---|---|---|---|
| 1 | 2 | 2 | 1 | 1 |
| 2 | 1 | 1 | 3 | 4 |
| 3 | 3 | 3 | 4 | 2 |
| 4 | 5 | 5 | 2 | 3 |
| 5 | 4 | 4 | 6 | 7 |
| 6 | 6 | 6 | 7 | 5 |
| 7 | 7 | 7 | 5 | 6 |

# Todd-Coxeter coset enumeration

Let $G = \langle X \mid R \rangle$, and $H = \langle h_1, \ldots, h_k \rangle \leq G$. The Todd-Coxeter coset enumeration procedure constructs a coset table for the action of $G$ on $H \backslash G$ as follows:

each coset has a **name**, consisting of a number and a word over $X^{\pm}$,

we start with an empty table, create new names as we need them, draw conclusions using the facts that

- all cosets are fixed by multipication by elements of $R$,
- the coset $H$ is fixed by multiplication by any $h_i$,

and hope that the process completes with some finite number of rows, which we then identify as $|G : H|$.

# Todd-Coxeter enumeration: an example

$G = \langle a, b \mid a^3, b^3, abab \rangle$, $\quad H = \langle a \rangle$. $H$ is the first coset; we make the first **deductions**.

| No. | Coset | $a$ | $a^{-1}$ | $b$ | $b^{-1}$ |
|---|---|---|---|---|---|
| 1 | $H$ | 1 | 1 | | |

**Events:**
Ded: $a \in H \Rightarrow 1a = 1$,
(equivalently $1a^{-1} = 1$).

We **define** the second coset $Hb$. $2 := 1b$, equivalently $2b^{-1} = 1$.

| No. | Coset | $a$ | $a^{-1}$ | $b$ | $b^{-1}$ |
|---|---|---|---|---|---|
| 1 | $H$ | 1 | 1 | 2 | |
| 2 | $Hb$ | | | | 1 |

**Events:**
Ded: $a \in H \Rightarrow 1a = 1$
Def: $2 := 1b$, $1 = 2b^{-1}$

# Todd-Coxeter enumeration: an example (2)

$G = \langle a, b \mid a^3, b^3, abab \rangle, \quad H = \langle a \rangle.$
We make a new definition $3 := 2b = Hbb$, equivalently $3b^{-1} = 2$.

| No. | Coset | $a$ | $a^{-1}$ | $b$ | $b^{-1}$ |
|---|---|---|---|---|---|
| 1 | $H$ | 1 | 1 | 2 | |
| 2 | $Hb$ | | | 3 | 1 |
| 3 | $Hbb$ | | | | 2 |
| | | | | | |

**Events:**
Ded: $a \in H \Rightarrow 1a = 1$
Def: $2 := 1b$, $1 = 2b^{-1}$
Def: $3 := 2b$

$b^3 \in R \Rightarrow Hb^3 = H \Rightarrow 3b = 1$, $1b^{-1} = 3$.

| No. | Coset | $a$ | $a^{-1}$ | $b$ | $b^{-1}$ |
|---|---|---|---|---|---|
| 1 | $H$ | 1 | 1 | 2 | 1 |
| 2 | $Hb$ | | | 3 | 1 |
| 3 | $Hbb$ | | | 1 | 2 |
| | | | | | |

**Events:**
Ded: $a \in H \Rightarrow 1a = 1$
Def: $2 := 1b$, $1 = 2b^{-1}$
Def: $3 := 2b$
Ded: $b^3 \in R \Rightarrow 1 = 3b$

# Todd-Coxeter enumeration: an example (3)

$G = \langle a, b \mid a^3, b^3, abab \rangle, \quad H = \langle a \rangle.$
$ababa \in R \Rightarrow 1abab = 1 \Rightarrow 2a = 1aba = 1b^{-1} = 3,\ 3a^{-1} = 2.$

| No. | Coset | $a$ | $a^{-1}$ | $b$ | $b^{-1}$ |
|---|---|---|---|---|---|
| 1 | $H$ | 1 | 1 | 2 | 3 |
| 2 | $Hb$ | 3 | | 3 | 1 |
| 3 | $Hbb$ | | 2 | 1 | 2 |
| | | | | | |

Ded: $a \in H \Rightarrow 1a = 1$
Def: $2 := 1b,\ 1 = 2b^{-1}$
Def: $3 := 2b$
Ded: $b^3 \in R \Rightarrow 1 = 3b$
Ded: $abab \in R \Rightarrow 3 = 2a$

Another new definition, $4 := 2a^{-1} = Hba^{-1}$, $4a = 2$.

| No. | Coset | $a$ | $a^{-1}$ | $b$ | $b^{-1}$ |
|---|---|---|---|---|---|
| 1 | $H$ | 1 | 1 | 2 | 3 |
| 2 | $Hb$ | 3 | 4 | 3 | 1 |
| 3 | $Hbb$ | | 2 | 1 | 2 |
| 4 | $Hba^{-1}$ | 2 | | | |
| | | | | | |

Ded: $a \in H \Rightarrow 1a = 1$
Def: $2 := 1b,\ 1 = 2b^{-1}$
Def: $3 := 2b$
Ded: $b^3 \in R \Rightarrow 1 = 3b$
Ded: $abab \in R \Rightarrow 3 = 2a$
Def: $4 := 2a^{-1}$

# Todd-Coxeter enumeration: an example (4)

$G = \langle a, b \mid a^3, b^3, abab\rangle, \quad H = \langle a\rangle.$

$a^3 \in R \Rightarrow 2a^3 = 2 \Rightarrow 2a^2 = 2a^{-1} \Rightarrow 3a = 4,\ 4a^{-1} = 3.$

| No. | Coset | $a$ | $a^{-1}$ | $b$ | $b^{-1}$ |
|---|---|---|---|---|---|
| 1 | $H$ | 1 | 1 | 2 | |
| 2 | $Hb$ | 3 | | 3 | 1 |
| 3 | $Hbb$ | 4 | 2 | 1 | 2 |
| 4 | $Hba^{-1}$ | 2 | 3 | | |

Ded: $a \in H \Rightarrow 1a = 1$

Def: $2 := 1b,\ 1 = 2b^{-1}$

Def: $3 := 2b$

Ded: $b^3 \in R \Rightarrow 1 = 3b$

Ded: $abab \in R \Rightarrow 3 = 2a$

Def: $4 := 2a^{-1}$

Ded: $2a^3 = 2 \Rightarrow 4 = 3a$

$abab \in R \Rightarrow 3abab = 3 \Rightarrow 3ab = 3b^{-1}a^{-1} \Rightarrow 4b = 4,\ 4b^{-1} = 4.$

| No. | Coset | $a$ | $a^{-1}$ | $b$ | $b^{-1}$ |
|---|---|---|---|---|---|
| 1 | $H$ | 1 | 1 | 2 | 3 |
| 2 | $Hb$ | 3 | 4 | 3 | 1 |
| 3 | $Hbb$ | 3 | 2 | 1 | 2 |
| 4 | $Hba^{-1}$ | 2 | 3 | 4 | 4 |

Ded: $a \in H \Rightarrow 1a = 1$

Def: $2 := 1b,\ 1 = 2b^{-1}$

Def: $3 := 2b$

Ded: $b^3 \in R \Rightarrow 1 = 3b$

Ded: $abab \in R \Rightarrow 3 = 2a$

Def: $4 := 2a^{-1}$

Ded: $2a^3 = 2 \Rightarrow 4 = 3a$

# Todd-Coxeter enumeration: an example (5)

$G = \langle a, b \mid a^3, b^3, abab \rangle$, $\quad H = \langle a \rangle$. Our coset table:

| No. | Coset | $a$ | $a^{-1}$ | $b$ | $b^{-1}$ |
|---|---|---|---|---|---|
| 1 | $H$ | 1 | 1 | 2 | 3 |
| 2 | $Hb$ | 3 | 4 | 3 | 1 |
| 3 | $Hbb$ | 3 | 2 | 1 | 2 |
| 4 | $Hba^{-1}$ | 2 | 3 | 4 | 4 |

is now **closed**, with 4 rows. So we have found a permutation rep. on 4 points; the subgroup $H$ fixes the first point. Since we have checked all the relations, we have found a group homomorphism form $G$ to $\mathcal{S}_4$.

Sometimes (not in this example) a deduction reveals a **coincidence** $i = j$ between two previously defined cosets. Two rows of the table are merged, further deductions or coincidences may be revealed. This computation can be huge, so the order in which info (deductions and concidences) is processed is crucial to its success. Different published **strategies** (by Felsch, or Haselgrove-Leech-Trotter) make different decisions here.

# Low index subgroups

Coset enumeration can be used to enumerate representatives of the conjugacy classes of all subgroups of $G$ of index up to a specified positive integer $n$. We simply build all coset tables with up to $n$ rows. This is known as the **low index subgroups algorithm**. Unlike coset enumeration itself, where the index of the subgroup is not known at the start, and indeed it is not known whether or not it is finite, the low index subgroups algorithm is guaranteed to complete. However its complexity appears to be worse than exponential in $n$.

# Subgroup presentation: theory

We have $G = \langle X \mid R \rangle$ and so $G \cong F/N$, where $F = F(X)$ is free and $N = \langle\langle R \rangle\rangle$.

If $H < G$, then $H \cong E/N$ for some $E$, with $N < E < F$. We can write $H = \nu(E)$, where $\nu : F \to G$, $\nu(g) = Ng$.

If $|G : H| < \infty$, then $|F : E| = |G : H| < \infty$, and $E = E(Y)$ for $|Y| < \infty$. So

$$F = \bigcup_{t \in T} Et, \quad G = \bigcup_{t' \in T'} Ht', \quad |T| = |T'|$$

We can choose the **transversal** $T$ st $1 \in T$. For $w \in F$, define $\overline{w} \in T$ to be the **coset representative** of $w$ in $T$, i.e. $w \in E\overline{w}$

**Theorem (Nielsen-Schreier)**

$$E = \langle Y \rangle, \text{ where } Y = \{tx(\overline{tx})^{-1} : t \in T,\, x \in X,\, tx \neq \overline{tx}\}.$$

*If $T$ is prefix-closed, then $Y$ freely generates $E$.*

# Subgroup presentation: Reidemeister-Schreier

Let $w = x_1 \cdots x_r$. Then $w \in Et$ for some $t$, and we can express it as a product $y_1 y_2 \cdots y_r t_r$ as follows. We apply **rewrites** of the form $tx \to y\overline{tx}$, $t \in T, x \in X^{\pm}$, $y \in Y$, working from the left, through a sequence:

$$1x_1 \cdots x_r \to y_1 t_1 x_2 \cdots x_r \quad \to y_1 y_2 t_2 x_3 \cdots x_r \to \cdots \quad \to y_1 y_2 \cdots y_r t_r \quad \text{where}$$

$$t_1 = \overline{1x_1}, y_1 = 1x_1 t_1^{-1}, \quad t_2 = \overline{t_1 x_2}, y_2 = t_1 x_2 t_2^{-1}, \quad t_3 = \overline{t_2 x_3}, y_3 = t_2 x_3 t_3^{-1},$$

$$\ldots\ldots\ldots\ldots\ldots\ldots \qquad t_r = \overline{t_{r-1} x_r}, \qquad y_r = t_{r-1} x_r t_r^{-1}.$$

Now denote each **right hand side** $y\overline{tx}$ by $\rho(tx)$, each product $y_1 y_2 \cdots y_r$ by $\rho(wt_r^{-1})$, and use the same notation for products over $T \cup X^{\pm}$ that are in $E$ and can be similarly rewritten (from the left). In particular we can rewrite conjugates of the elements of $R$, and we have:

**Theorem (Reidemeister-Schreier)**

$$H \cong \langle Y \mid \rho(twt^{-1}), \forall w \in R, t \in T \rangle$$

the basis for the Reidemeister-Schreier algorithm.

# The Reidemeister-Schreier algorithm: an example

If $|G : H| < \infty$ and we have a coset table for $H$ in $G$, then we can use the **Reidemeister-Schreier algorithm** to compute the Schreier generators and write down explicitly the presentation given by the R-S theorem.

For example: $G = \langle x, y \mid x^3,\, y^4,\, (xy)^2 \rangle$, $H = \langle x, yx^{-1}y^{-2} \rangle$, $|G : H| = 4$.

| Coset no. | $x$ | $y$ | $x^{-1}$ | $y^{-1}$ |
|---|---|---|---|---|
| 1 | 1 | $\underline{2}$ | 1 | 3 |
| 2 | $\underline{3}$ | $\underline{4}$ | 4 | 1 |
| 3 | 4 | 1 | 2 | 4 |
| 4 | 2 | 3 | 3 | 2 |

The first occurrence of each coset number in the table is underlined, called its **definition**. We use this info to choose $T$, here as $\{1,\, y,\, yx,\, y^2\}$.

# The Reidemeister-Schreier algorithm: an example (2)

The elements of $Y$ correspond to the non-trival products $tx(\overline{tx})^{-1}$. We define them by inserting new symbols (and their inverses) into those cells of the coset table that don't contain definitions (or their inverses).

| Coset no. | $x$ | $y$ | $x^{-1}$ | $y^{-1}$ |
|---|---|---|---|---|
| 1 | $a1$ | $\underline{2}$ | $a^{-1}1$ | $c^{-1}3$ |
| 2 | $\underline{3}$ | $\underline{4}$ | $d^{-1}4$ | 1 |
| 3 | $b4$ | $c1$ | 2 | $e^{-1}4$ |
| 4 | $d2$ | $e3$ | $b^{-1}3$ | 2 |

We've made some choices (e.g. which elements are in $Y$, which is $Y^{-1}$), and have chosen $Y = \{a, b, c, d, e\}$ with

$$a = x,\ b = yx^2y^{-2},\ c = yxy,\ d = y^2xy^{-1},\ e = y^3x^{-1}y^{-1}.$$

# The Reidemeister-Schreier algorithm: an example (3)

We calculate the relators $\rho(twt^{-1})$ of $H$ by tracing out the image of $t \in T$ under each relator $w \in R$, using the modified coset table; we must have $tw = t$. If $w = u^m$ and we know that $t_j = t_i u$, then we don't need to apply $w$ to $t_j$ as well as to $t_i$, i.e. that gives nothing new. So, below we only need to trace out $1x^3$ and $2x^3$, not $3x^3$ or $4x^3$.

| | $x$ | | $x$ | | $x$ | |
|---|---|---|---|---|---|---|
| 1 | | $a1$ | | $a^2 1$ | | $a^3 1$ |
| 2 | | 3 | | $b4$ | | $bd2$ |

| | $y$ | | $y$ | | $y$ | | $y$ | |
|---|---|---|---|---|---|---|---|---|
| 1 | | 2 | | 4 | | $e3$ | | $ec1$ |

| | $x$ | | $y$ | | $x$ | | $y$ | |
|---|---|---|---|---|---|---|---|---|
| 1 | | $a1$ | | $a2$ | | $a3$ | | $ac1$ |
| 3 | | $b4$ | | $be3$ | | $beb4$ | | $(be)^2 3$ |
| 4 | | $d2$ | | $d4$ | | $d^2 2$ | | $d^2 4$ |

# The Reidemeister-Schreier algorithm: an example (4)

We extract the relators from the right hand columns of those tables, and so deduce

$$H \cong a, b, c, d, e \mid a^3,\ bd,\ ec,\ ac,\ (be)^2,\ d^2\rangle.$$

We apply Tietze transformations to eliminate the generators $e$ $(= c^{-1})$, $d$ $(= b^{-1})$, and then $c$ $(= a^{-1})$, and derive

$$H \cong \langle a, b, c, d, e \mid a^3,\ (ab)^2,\ b^2\rangle;$$

we recognise that $H \cong \mathcal{S}_3$.

The procedure can be modified to derive presentations on **user supplied generators**. But in this case calculation of the presentation needs to be performed during the coset enumeration; this can result in very long relators.

# Contents: Lecture 2

4. Rewriting to solve WP(G): Dehn, van Kampen, Knuth-Bendix
5. Introducing automatic, biautomatic and hyperbolic groups
6. Computation in automatic and biautomatic groups
   - Using the FSA for computation
   - Construction and verification of automatic structures
7. Computation in hyperbolic groups
   - Cone types and geodesics
   - Proving hyperbolicity
   - Conjugacy problem in hyperbolic groups

# Dehn's solution to $\mathtt{WP}(G)$ for a surface group $G$

Dehn described a solution to the word problem for

$$\pi_1(S_g) = \langle a_1, b_1, a_2, \ldots, b_g \mid r := a_1 b_1 a_1^{-1} b_1^{-1} \cdots a_g b_g a_g^{-1} b_g^{-1} = 1 \rangle,$$

the fundamental group of an orientable surface of genus $g$.

We call a word that's a cyclic permutation of $r$ or its inverse $r^{-1}$ a **symmetrised relator**, $\hat{R}$ the set of those.

For $g > 1$, any word that represents the identity in $\pi_1(S_g)$ must contain a subword $u$ that is more than half of a symmetrised relator.

## Algorithm (Dehn's solution for the word problem in surface groups)

While $w$ contains a subword $u$ that is more than half of a (symmetrised) relator $uv^{-1}$, replace $u$ in $w$ by $v$, and repeat.
If $w \to \epsilon$ (the empty word), then return '$w =_G 1$'; otherwise return '$w \neq_G 1$'.

# Applying Dehn's algorithm in $\pi_1(S_2)$

In $\pi_1(S_2)$,

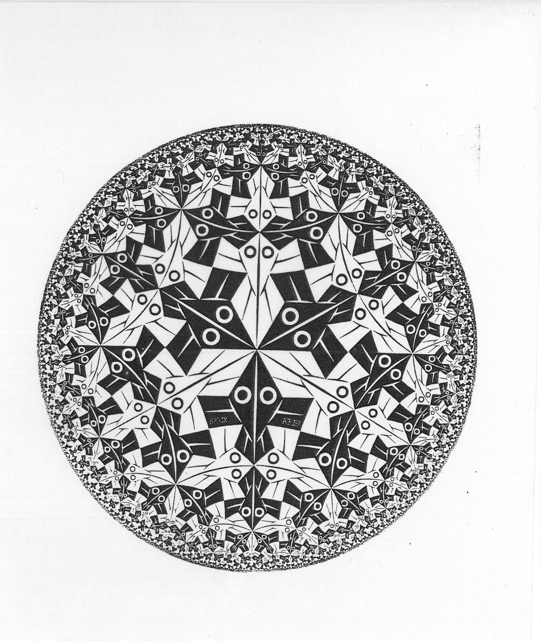
$$\text{where} \quad dcd^{-1}c^{-1}bab^{-1}a^{-1} = 1 \quad \text{and} \quad b^{-1}a^{-1}dcd^{-1}c^{-1}ba = 1,$$

as conjugates of the inverse of $aba^{-1}b^{-1}cdc^{-1}d^{-1}$, the product $w_1 = b^{-1}a^{-1}dccd^{-1}c^{-1}bab^{-1}a^{-1}c^{-1}ba$ is proved trivial since

$$b^{-1}a^{-1}dc\color{red}{cd^{-1}c^{-1}bab^{-1}a^{-1}}\color{black}{c^{-1}ba} \quad \to \quad \color{green}{b^{-1}a^{-1}dcd^{-1}c^{-1}ba}\color{black}{} \to \epsilon,$$

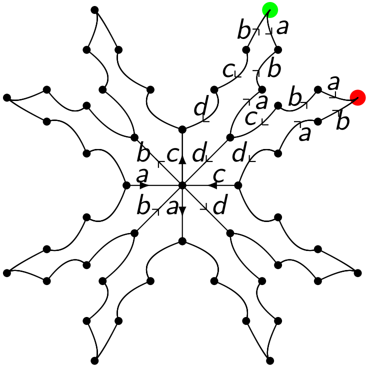
But the product $w_2 = b^{-1}a^{-1}dccd^{-1}c^{-1}b$ is proved non-trivial, since it does not reduce to $\epsilon$; in fact it does not reduce at all. Dehn's algorithm works because the **Cayley graph** of the surface group inherits geometry from the negatively curved Poincaré disc, in which it embeds.

# The geometry of the Poincaré disc



# The geometry behind Dehn's algorithm

The diagram below shows the neighbourhood of 1 in $\mathcal{G}(\pi_1(S_2))$. A word $w_0$ labels the outer boundary. Dehn's algorithm reduces $w_0$ to $\epsilon$ using 8 length reducing substitutions.




$w_0 = a^{-1}b^{-1}ca \cdots ab$,

length 48,

We can also see the words:

$w_1 = b^{-1}a^{-1}dcc \cdots ba$,

length 14,

$w_2 = b^{-1}a^{-1}dccd^{-1}c^{-1}b$,

length 8,

Each of those substitutions corresponds to **peeling off** a **relation cell** from the loop enclosed by the current word $w$. The negative curvature of the Cayley graph guarantees that we always have a cell with more than half of its edges on the boundary of the current loop.

# Dehn's algorithm only works for 'word hyperbolic' groups $G$

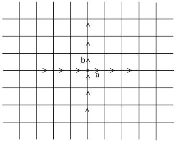
Dehn's algorithm solves $\mathtt{WP}(G) \iff \mathcal{G}(G)$ is a hyperbolic space $\iff G$ is a **word hyperbolic** group. Otherwise we need a different strategy.

But more generally we can often use the geometry of $\mathcal{G}(G)$, or of other spaces on which $G$ acts, to help us answer various questions about $G$,

- to solve decision problems such as the word problem, conjugacy problem, or other equations,
- to answer some questions about finite order (of the group, or of its elements),
- to understand the structure of **geodesic** words (shortest products representing an element), or other **normal forms**.

Maybe we can prove the group **automatic** or **biautomatic** (in which case it has a particularly well structured normal form); computation with (bi)automatic groups is particularly straightforward.

# Solving the word problem in abelian groups




E.g. $\mathbb{Z}^2 = \langle a, b \mid ba = ab \rangle$

From the equation $ba = ab$, we can deduce 3 others:

$$ba^{-1} = a^{-1}b, \quad b^{-1}a = ab^{-1}, \quad b^{-1}a^{-1} = a^{-1}b^{-1}.$$

Now, given a product $w$ of positive and negative powers of $a$ and $b$, we just keep applying rules:

$$ba \to ab, \quad ba^{-1} \to a^{-1}b, \quad b^{-1}a \to ab^{-1}, \quad b^{-1}a^{-1} \to a^{-1}b^{-1}$$

until $w$ has been transformed to the form $a^i b^j$. The original product $w$ represents the identity iff we reach the empty word $\epsilon$, with $i = j = 0$.

Geometrically, we've converted the path $\gamma$ in the Cayley graph that's labelled by $w$ and starts at 1 into a path $\gamma'$ between the same two vertices as $\gamma$, but which does all its movement in the $a$ direction before all its movement in the $b$ direction. Each reduction corresponds to pulling the path across a square that contains two or more of its edges.

# Solving the word problem in nilpotent groups

E.g. $\quad G = \langle a, b, c \mid ba = abc,\ ca = ac,\ cb = bc \rangle$ (integer Heisenberg gp)

From the defining relations, we deduce additional equations

$$ba^{-1} = a^{-1}bc^{-1}, \quad b^{-1}a = ab^{-1}c^{-1}, \quad b^{-1}a^{-1} = a^{-1}b^{-1}c,$$
$$c^{-1}a = ac^{-1},\ c^{-1}b = bc^{-1},\ c^{\pm 1}a^{-1} = a^{-1}c^{\pm 1},\ c^{\pm 1}b^{-1} = b^{-1}c^{\pm 1}$$

Now, given any product of positive and negative powers of $a$, $b$ and $c$, we keep applying those equations (replacing left hand sides by right hand sides) to the product to get it into the form $a^{\pm i}b^{\pm j}c^{\pm k}$. A product representing the identity element must rewrite to the empty word $\epsilon$.

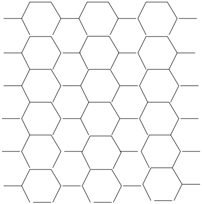
Application of these rules solves the word problem in this group in cubic time, and a similar strategy solves $\mathtt{WP}(G)$ in any nilpotent group $G$ in polynomial time. But it's harder in this example to see guidance from the geometry of $\mathcal{G}(G)$.

# Solving the word problem in Coxeter groups

A Coxeter group on $x_1, \ldots, x_n$ is presented by relations $x_i^2 = 1$, $\forall i$, together with some braid relations $x_i x_j x_i \cdots = x_j x_i x_j \cdots$ (relating two words length $m_{ij}$),

E.g. $\langle a, b, c \mid a^2 = b^2 = c^2 = 1, aba = bab, bcb = cbc, aca = cac \rangle$,

whose Cayley graph tesselates the plane with regular hexagons.



We can reduce any (positive) word $w$ to a geodesic rep. by some sequence of replacements of subwords equal to one side of a braid relation by the other side, combined with deletion of any subwords $x_i^2$; i.e. we shorten a word by pulling across hexagons some sections of length $\geq 3$ of the corresponding path.

The same process gives an exponential time solution to the word problem in any Coxeter group. It's slow because it's unclear which braid relations to apply (we could apply any relation in either direction).

# In general, solving $\mathtt{WP}(G)$ is not straightforward

The solutions we've seen so far use rewriting techniques, and the solutions for hyperbolic, abelian and Coxeter groups are guided by the geometry of the Cayley graph.

But in general solving the word problem is not easy, and may be impossible.

**E.g.** so far as I know, no solution is known for the word problem of the following Artin group:

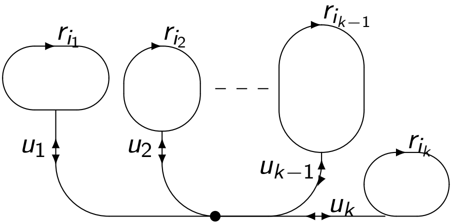
$$\begin{aligned} \langle a, b, c, d \quad | \quad & aba = bab, bcb = cbc, aca = cac, \\ & cdc = dcd, dbd = bdb, ad = da \rangle. \end{aligned}$$

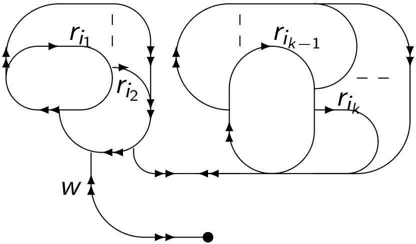
However, rewriting systems for Artin groups of spherical and FC types, and for those of sufficiently large type, are well known and well studied.

# van Kampen diagrams

Let $G = \langle X \mid R \rangle$ and suppose that $w \in (X^{\pm})^*$ is freely reduced. Then $w \in \mathtt{WP}(G, X) \iff w =_{\mathbb{F}(X)} u_1 r_{i_1} u_1^{-1} \cdots u_k r_{i_k} u_k^{-1}$, where $u_1, \ldots, u_k \in (X^{\pm})^*$, $r_{i_1}, \ldots, r_{i_k}$ are relators (or inverses of such) from $R$.

Within a plane, we draw $k$ distinct paths labelled $u_1, \ldots, u_k$ from a basepoint, and at the end of the $j$-th path we attach a loop labelled $r_{i_j}$. Working out from the basepoint, we identify any two edges with the same source and same label that are adjacent within the plane. The resulting planar diagram $\Delta$ has boundary labelled by $w$, and decomposes into $k$ regions, each with boundary labelled by a relator. We call it a **van Kampen diagram of area $k$ for $w$**.



# The Dehn function of a group

Area($w$) is defined to be the min. area of any van Kampen diag. $\Delta$ for $w$.

We define the **isoperimetric function**, or **Dehn function**, $f_G(n)$ of $G$ by

$$f_G(n) := \max\{\mathrm{Area}(w) : w \in \mathtt{WP}(G,X),\ |w| \leq n\}.$$

Given $f_G(n)$, we can decide in time $\leq \exp(Kf_G(n))$ if an input $w$ is in $\mathtt{WP}(G,X)$, by enumerating and freely reducing all candidate **factorisations** $u_1 r_{i_1} u_1^{-1} \cdots u_k r_{i_k} u_k^{-1}$ with $|u_i| \leq (|w| + i \max\{|r|; r \in R\})/2, \quad k \leq f_G(n)$.

Without $f_G(n)$ we can still enumerate all possible factorisations of $w$, but may not know when to stop searching.

If $H = \langle Y \mid S \rangle \cong G$, or $H \subseteq G$ with $|G : H| < \infty$, then $f_G(n) \approx f_H(n)$, i.e.
$\exists C,\ f_H(n) \leq Cf_G(Cn+C)+Cn+C, \quad \exists D,\ f_G(n) \leq Df_H(Dn+D)+Dn+D$,
We say that $f_G$ **dominates** $f_H$ ($f_H \preceq f_G$) and $f_H$ **dominates** $f_G$.

In this case, $\mathtt{WP}(G,X)$ is soluble $\iff$ $\mathtt{WP}(H,Y)$ is soluble.

# Rewrite systems

Some of the solutions we have described to $\mathtt{WP}(G, X)$, e.g. those with $G$:

- a surface group or any word hyperbolic group (Dehn's algorithm),
- $\mathbb{Z}^2 = \langle a, b \mid ab = ba \rangle$,
- the Heisenberg group $\langle a, b, c \mid ba = abc,\ ca = ac,\ cb = bc \rangle$.

use **rewrite systems**, that reduce an input word $w$ to $\epsilon \iff w =_G 1$.

We define a rewrite system (RWS) $\mathcal{R}$ over an alphabet $A$ to be a set of substitution rules $\rho : u \to v$, for $u, v \in A^*$ (plus instructions on any restrictions on their application); if $w$ is a string over $A$ then an application of $\rho$ to $w$ replaces a substring $u$ of $w$ by the string $v$, to derive a word $w'$ We write $w \xrightarrow{\mathcal{R}} w'$ or $w \to w'$, and if $w \to \cdots \to w_n$, we write $w \to^* w_n$.

For the three examples above we have:

- $\mathcal{R} = \{u \to v : (|u| > |v|, uv^{-1} \in \hat{R}\}$,
- $\mathcal{R} = \{b^\eta a^\epsilon \to a^\epsilon b^\eta : \epsilon, \eta \in \{\pm 1\}\}$,
- $\mathcal{R} = \{b^\eta a^\epsilon \to a^\epsilon b^\eta c^{\epsilon\eta},\ c^\eta a^\epsilon \to a^\epsilon c^\eta,\ c^\eta b^\epsilon \to b^\epsilon c^\eta : \epsilon, \eta \in \{\pm 1\}\}$

In each case, there are no restrictions on application of rules.

# Rewrite systems (2)

Let $\preceq$ be a partial order on the set $A^*$ of words over $A$, satisfying

$$\forall w \in A^*, \quad (v \preceq u \Rightarrow (wv \preceq wu) \wedge (vw \preceq uw)).$$

We say that a RWS $\mathcal{R}$ is **compatible with $\preceq$** if

$$(u \to v) \in \mathcal{R} \Rightarrow v \preceq u.$$

The three RWS on the previous slide are compatible with (respectively):
- ordering by word length,
- the **shortlex** ordering over $\{a^{\pm}, b^{\pm}\}$, where $v <_{\mathrm{slex}} u$ if $|v| < |u|$ or if $|v| = |u|$ but $v$ precedes $u$ lexicographically (given $a < a^{-1} < b < b^{-1}$),
- a **wreath product** of the shortlex orders over $\{a^{\pm}, b^{\pm}\}$ and $\{c^{\pm}\}$, for which we order two words first via their projections onto $\{a^{\pm}, b^{\pm}\}$ and then via the sequences of words over $c^{\pm}$ between symbols from $a^{\pm}, b^{\pm}$.

The 2nd and 3rd of these are total orders.

# Confluence and completeness

A RWS $\mathcal{R}$ over an alphabet $A$ is
**Noetherian** if all chains of strings $w \to w_1 \to \ldots \to w_n \to \ldots$ are finite,
**confluent** if $(w \to^* w_1) \wedge (w \to^* w_2) \Rightarrow \exists w'((w_1 \to^* w') \wedge (w_2 \to^* w'))$,
**complete** if Noetherian and confluent,
**locally confluent** if $(w \to w_1) \wedge (w \to w_2) \Rightarrow$
$$\exists w'((w_1 \to^* w') \wedge (w_2 \to^* w')).$$
In fact a RWS is also complete if Noetherian and locally confluent.

We see that our RWS for $\pi_1(S_2)$ is Noetherian but not confluent, since $bab^{-1}a^{-1}dc^{-1}d^{-1}ab \to$ either $cdc^{-1}c^{-1}d^{-1}ab$ or $bab^{-1}a^{-1}c^{-1}ba$, but then no further. Our other two examples are complete.

The **Knuth-Bendix** (KB) procedure takes as input a finite RWS $\mathcal{R}$ compatible with a total order $\preceq$ and attempts to build a finite complete system $\hat{\mathcal{R}}$, for which $(w \xrightarrow{\mathcal{R}}{}^* w') \Rightarrow (w \xrightarrow{\hat{\mathcal{R}}}{}^* w')$,
by adding rules $w_1 \to w_2$ or $w_2 \to w_1$ (compatible with $\preceq$) when confluence fails. It is not guaranteed to terminate.

# Using a complete RWS to solve $\mathtt{WP}(G)$

Let $G = \langle X \mid R \rangle$, $\preceq$ be a total order st $v \preceq u \Rightarrow vw \preceq uw$, $wv \preceq wu$, and $\mathcal{R} = \{u \to v : v \preceq u, uv^{-1} \in \hat{R}\} \cup \{aa^{-1} \to \epsilon : a \in X^{\pm}\}$.

If $\mathtt{KB}$ derives a finite complete RWS $\hat{\mathcal{R}}$ from $\mathcal{R}$, then application of $\hat{\mathcal{R}}$ solves $\mathtt{WP}(G, X)$ in time $O(f_G(n))$.

**Proof:** If $w \neq \epsilon$ and $w =_G 1$ then $w =_{\mathbb{F}(X)} u_1 r_{i_1} u_1^{-1} \cdots u_k r_{i_k} u_k^{-1}$ and we can find $w_1, \ldots, w_r$ (with $r$ odd, but possibly $w = w_1, w_r = \epsilon$) with

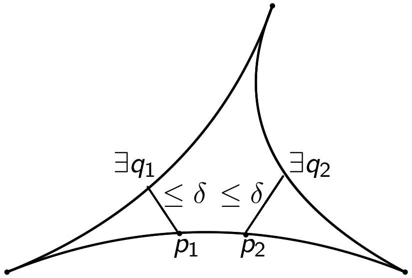
$$w \xrightarrow{\hat{\mathcal{R}}}{}^{*} w_1 \; {}^{*}\xleftarrow{\hat{\mathcal{R}}} \cdots \xrightarrow{\hat{\mathcal{R}}}{}^{*} w_{i-1} \; {}^{*}\xleftarrow{\hat{\mathcal{R}}} w_i \xrightarrow{\hat{\mathcal{R}}}{}^{*} w_{i+1} \; {}^{*}\xleftarrow{\hat{\mathcal{R}}} \cdots \xrightarrow{\hat{\mathcal{R}}}{}^{*} w_r = \epsilon.$$

If $r \geq 3$, then completeness implies we can find such a sequence of length $r - 2$, so ultimately one of length 1, and hence we find $w_1', \ldots, w_s'$ st $w \to w_1' \to \cdots \to w_s' = \epsilon$. The connection between the rewrite sequence and the factorisation of $w$ gives us a bound on the total number $s$ of rewrites from $\hat{R}$, in terms of $f_G(n)$.

# Rewriting to solve WP(G,X) when $G$ is hyperbolic

In general, the RWS for Dehn's algorithm is not complete. But it solves $\mathtt{WP}(G, X)$ in linear time in any (word) hyperbolic group.

$G$ is hyperbolic $\iff$ Dehn's algorithm solves $\mathtt{WP}(G)$ $\iff$ $f_G(n)$ is linear.
Equivalently, $G$ is **(word) hyperbolic** if its Cayley graph $\mathcal{G}(G, X)$ is $\delta$-hyperbolic for some $\delta$, ie any triangle of geodesics is $\delta$-slim.




The definition formalises properties found in (hyperbolic) surface groups, or in any $\pi_1(M)$, $M$ compact hyperbolic.

$\mathbb{F}_n$ is hyperbolic; $\mathbb{Z}^n$ is not, no hyperbolic group can even contain $\mathbb{Z}^2$.

# Solving WP(G,X) when $G$ is a Coxeter group

We've already described a straightforward algorithm that solves the word problem in any Coxeter group.The algorithm is easy to describe but rather slow (running in exponential time), since we have to apply all possible sequences of braid relations to a word to be sure that it admits no reduction.

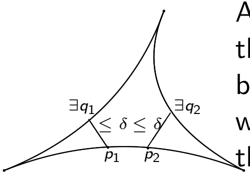
But in fact every Coxeter group is automatic (Brink&Howlett,1993), and hence its word problem can be solved in quadratic time (as we shall see later).

And many (but not all) Coxeter groups are known to have finite, complete rewrite systems (Hermiller,1994).

And all Coxeter groups have soluble conjugacy problem. But despite that (see later), it's not known whether they are all biautomatic.

# Introducing hyperbolic, automatic and biautomatic groups

The features that Dehn identified in surface groups that made their word problems easy to solve are found more generally in all word hyperbolic groups.




As well as allowing easy solution of the word problem, the negative curvature of the Cayley graph of a hyperbolic group $G$ allows various features of geodesic paths within those graphs to be represented by $\mathrm{FSA}$, which then facilitate computation with the groups.

And the same methods can work more generally.

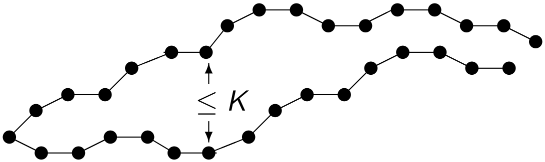
Automatic and biautomatic groups were introduced (by Thurston et al.) as a generalisation of hyperbolic groups, with many of the same algorithmic properties.

All hyperbolic groups are both automatic and biautomatic.

# Defining automatic and biautomatic groups

$G = \langle X \rangle$ is **automatic** if $\exists$ a set of paths from the vtx 1 in $\mathcal{G}(G, X)$, a corresponding set of words $L \twoheadrightarrow G$ labelling them, and a constant $K$, st

- $L$ is a **regular set** (i.e. can be recognised by a finite state automaton with alphabet $X^{\pm}$)
- if $v, w \in L$ satisfy $v =_G w$ or $vx =_G w$ for $x \in X$, then the corresponding paths $\gamma_1(w), \gamma_1(v)$ **$K$-fellow travel**,

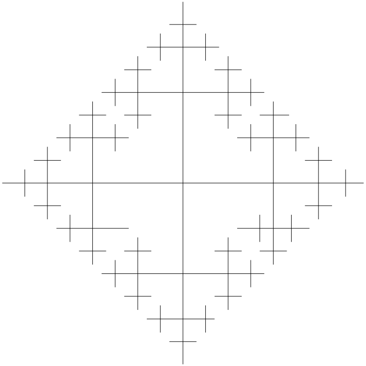


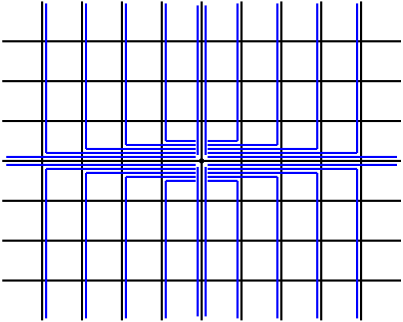

and is **biautomatic** if in addition

- the paths $\gamma_1(w)$ $\gamma_x(v)$ traced out from 1 and $x$ by words $w$ and $v$ that satisfy $xv =_G w$ must $K$-fellow travel.

Without the restriction on regularity of $L$, $G$ is (bi)combable.

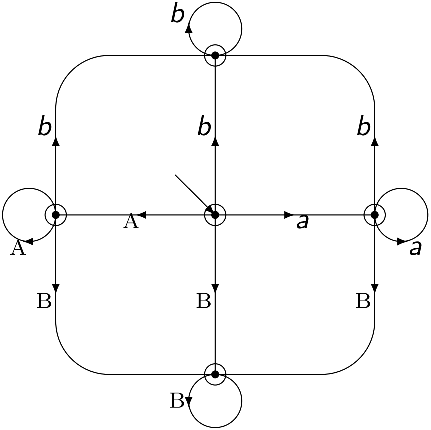
# $F_2$ and $\mathbb{Z}^2$ are biautomatic, and $F_2$ is hyperbolic.





In $F_2$ we select all geodesic paths/words. But in $\mathbb{Z}_2$ we have to restrict to a subset, such as $\{a^i b^j : i, j \in \mathbb{Z}\}$ (the 'shortlex' language), in order to get fellow travelling; note that geodesics $a^i b^i$ and $b^i a^i$ diverge to distance $i$. Similarly, geodesic triangles in $\mathcal{G}(F_2)$ are 0-slim, but those in $\mathcal{G}(\mathbb{Z}^2)$ are fat. In general, $G(X)$ is **shortlex automatic** if the shortlex language is the language of an automatic structure.

# Word acceptor $\mathcal{W}$ for $\mathbb{Z}^2$, accepting $\{a^i b^j\}$




This FSA has **six** states, but we can only see **five**. All transitions that are not shown are to that sixth (non-accepting) **failure** state, and all transitions from it return to it.

The other five states are all accepting states (and so ringed).

# What's the use of hyperbolicity, (bi)automaticity?

- Hyperbolic groups are very easy to compute with. Dehn's algorithm solves the word problem in any hyperbolic group in linear time.
- Automatic groups are all `fp`, have word problem soluble in quadratic time. Biautomatic groups have soluble conjugacy problem: $?\exists g,\ u_1^g =_G u_2$.
- Use of the word acceptor FSA of an automatic group allows tests for some basic properties (e.g. finiteness).
- The fellow traveller (`ft`) condition satisfied by an automatic group can be expressed in terms of FSA called multiplier automata. Various algorithms can de described in terms of computations with those FSA.

**Examples:** A group $G$ is hyperbolic $\iff$ its set $\mathrm{Geo}(G)$ of all geodesics gives (bi)automatic structure. Additionally, we have $\pi_1(\mathcal{X})$ for many compact 3-manifolds $\mathcal{X}$, all Coxeter groups, many Artin groups, mapping class groups of all surfaces of finite type. And the class is closed under various group operations (direct and free product, extensions by finite groups (certain) HNN extensions.

# Formulation of automaticity via 1- and 2-string FSA

$G = \langle X \rangle$ is **automatic** if $\exists$:
a set $L$ of words $L$ containing at least one rep. of each group element,
an FSA $\mathcal{W}$ (the **word acceptor**) that reads strings over $X^{\pm}$,
and FSA $\mathcal{M}_g$ for each $g \in X^{\pm} \cup \{1\}$ (the **multiplier automaton**), that read strings over $X^{\pm} \times X^{\pm}$(or equivalently pairs of words $v, w$ over $X$), st

- $\mathcal{W}$ recognises the set $L$,
- $\mathcal{M}_g$ recognises the set or pairs $(v, w)$ of words over $X$, for which $v, w \in L$ and $vg =_G w$.

We can construct a **difference machine** $\mathcal{D}$ that recognises pairs $(v, w)$ of words over $X$ that $K$-fellow travel. For each $g$ we construct $\mathcal{M}_g$ to accept $(v, w)$ iff $v, w \in L$, $(v, w) \in L(\mathcal{D})$ and $v^{-1}w =_G g$.

Formulation of the fellow traveller property in terms of FSA can make computations easier to describe. Some of what I'll describe has been programmed within kbmag (Holt), more could be programmed using it.

# Building composite FSA

When $G$ is an automatic group, standard operations on FSA build various FSA from the FSA of the automatic structure for $G$, that can be used in computation with $G$.

Given FSA $M_1, M_2$, it's straightforward to build FSA $M^\wedge$ and $M^\vee$ with languages that are the intersection, union of the languages of $M_1, M_2$.

Similarly we can apply other combinations of boolean and logical operators to FSA.

Then, given the multiplier automata $\mathcal{M}_g$ for an automatic group $G$ (for generators $g$) we can construct a **multiplier automaton** $\mathcal{M}_u$, for a word $u$ over $X$, recognising pairs of words $(v, w)$ with $w =_G vu$ and $(v, w)$ fellow travelling (at distance $|u|K$). If $G$ is biautomatic, then we can similarly define and construct left multipliers ${}_g\mathcal{M}$ and ${}_u\mathcal{M}$.

## Reduction to $L$, solution of $\mathtt{WP}(G)$ run in quadratic time

Given $w = a_1 \cdots a_n$, we can find a rep. $u \in L$ of $w$ in time $Cn^2$, some $C$.

$v \in L$ represents $a_1 \cdots a_{k-1}$, $(v, v') \in L(\mathcal{M}_{a_k}) \Rightarrow v' \in L$ represents $va_k$.
Given $v$, we can find $v'$ of length at most $|v| + E$ in time at most $D|v|$, where $D, E$ are constants depending only on the FSA $\mathcal{M}_{a_k}$.
We find $u$ by iterating this process $n$ times; the $k$-th step produces a word in $L$ of length at most $A + Bk$ representing $a_1 \ldots a_k$, in time at most $Ck$ for some $C$. Hence we find a representative of $u$ in time at most $Cn^2$.

Given $u_0 \in L$ representing 1, we can use $\mathcal{M}_1$ to check in time $O(|u|)$ whether or not $u =_G u_0$, and hence

If $G$ is automatic, $\mathtt{WP}(G)$ can be solved in quadratic time.

Where $H < G$, and the coset system $(G, H)$ is **coset automatic** ($\exists$ regular set of coset reps with similar `ft` propeties (Redfern; Holt&Hurd)), we have an $O(n^2)$ algorithm for subgroup membership.

# Using FSA to solve $\mathtt{CP}(G)$ for biautomatic $G$

When $G$ is biautomatic then, for all $u_1, u_2 \in (X\pm)^*$ the set of conjugating words

$$\{w \in (X^{\pm})^* : wu_1 =_G u_2 w\}$$

is a regular set, the language of an FSA that can be built out of the FSA of the biautomatic structure for $G$ (Gersten&Short,1991).

Specifically, the method that constructs general multipliers allows us also to construct an FSA accepting

$$\{(v, w) : v, w \in L, vu_1 =_G u_2 w\}.$$

It is now elementary to use the $\mathcal{M}^\wedge$ construction to construct from this and the 'diagonal' of $\mathcal{W}$, an FSA accepting the language of conjugators.

So $\mathtt{CP}(G)$ can be solved using a combination of operations on FSA.

NB: Short proved geometrically that any bicombable group $G$ has soluble $\mathtt{CP}(G)$, but we need regularity of $L$ for this particular construction.

# Construction of shortlex automatic structures

Starting point, a finite set of rewrite rules for $G = G(X)$, derived from presentation, compatible with shortlex (wrt some ordering of $X$).

- Run `KB` for a while to get a RWS $\mathcal{R}$ and construct an associated **word difference machine** $\mathcal{D}$ accepting pairs $(u, v)$ for which $(u \to v) \in \mathcal{R}$
- Construct word acceptor $\mathcal{W}$ and multiplier automata $\mathcal{M}_g$ from $\mathcal{D}$. If preliminary checks on those automata fail, then restart `KB` and rebuild $\mathcal{D}$ and other automata. Repeat as necessary until checks pass.
- Verify correctness using **axiom checking**: checks based on construction of various automata using logical operations on $W$ and the multipliers $\mathcal{M}_g$. If these checks fail, give up.

If procedure terminates, after successful axiom checking $G$ is proved shortlex automatic with word acceptor $\mathcal{W}$ and multipliers $\mathcal{M}_g$, Otherwise procedure may loop endlessly, or give up due to lack of time or space.

# Hyperbolic $G$: Cone types and geodesics

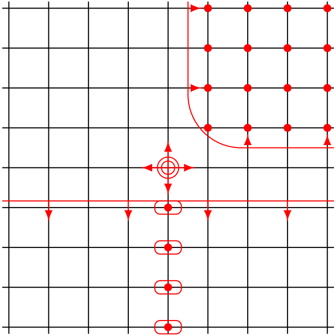
$G(X)$ is hyperbolic $\iff$ the set $\text{Geo}(X)$ of geodesic words over $X$ is the language of an automatic structure $\iff \cdots\cdots$ a biautomatic structure.

In particular, in any hyperbolic group, and for some $X$, in some (not nec. automatic) other groups, **e.g.** $\mathbb{Z}^n$, Coxeter groups, $\text{Geo}(X)$ is regular;

equivalently $G$ has finitely many **cone types**, equiv. classes of $\approx$ on $G$, where $g \approx h$ iff, for geodesic reps. $w_g, w_h$, $\forall w \in (X^{\pm})^*$,

$$w_g w \text{ is geodesic} \iff w_h w \text{ is geodesic.}$$

We visualise the cone type of $g$ as the sector of $\mathcal{G}(G)$ containing those words $w$ that continue $w_g$ geodesically; in $\mathbb{Z}^2$ there are 9 (we show 3 of those):



# Proving hyperbolicity

Recall: $G(X)$ is hyperbolic $\iff \exists$ automatic structure on $\mathrm{Geo}(X)$
$\Rightarrow G(X)$ is shortlex automatic, wrt any ordering of $X$.

Suppose that $G$ **is** hyperbolic, and that $\mathcal{W}, \mathcal{D}_{\mathcal{W}}$ are the word acceptor, and difference machine of a shortlex automatic structure over $X$.

Then $\exists$ a set of word differences, and associated difference machine $\mathcal{D}_{\mathrm{Geo}}$ (probably bigger than $\mathcal{D}_{\mathcal{W}}$) for which $\mathrm{Geo}(X)$ is equal to its (regular) subset

$$\mathrm{FtGeo}(\mathcal{W}, \mathcal{D}) := \{v : \exists w \in L(\mathcal{W}), (v, w) \in L(\mathcal{D}^{\epsilon}), |w| = |v|\}.$$

We could verify that $G$ is hyperbolic if we could find $\mathcal{D}^{\mathrm{Geo}}$, construct the set $\mathrm{FtGeo} := \mathrm{FtGeo}(\mathcal{W}, \mathcal{D}^{\mathrm{Geo}})$, and then prove that $\mathrm{FtGeo} = \mathrm{Geo}(X)$.

We need our method to terminate with output $(\mathcal{D}^{\mathrm{Geo}}, \mathrm{FtGeo})$ when $G$ is hyperbolic, but to fail in some way when it is not.

Then we have a valid test for hyperbolicity.

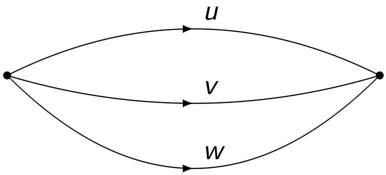
# A procedure to verify hyperbolicity

A looping procedure builds successive machines $\mathcal{D}_i$, $i \geq 0$ from which we construct FSA accepting the subsets $\mathrm{FtGeo}_i := \mathrm{FtGeo}(\mathcal{W}, \mathcal{D}_i)$ of $\mathrm{Geo}(X)$. We start with $\mathcal{D}_0 = \mathcal{D}_{\mathcal{W}}$, and $\mathrm{FtGeo}_0 = \mathrm{FtGeo}(\mathcal{W}, \mathcal{D}_0)$.

Hyperbolicity is verified if $\mathcal{D}_i \to \mathcal{D}_{\mathrm{Geo}}$, in which case $\mathrm{FtGeo}_i \to \mathrm{Geo}(X)$.

At each stage $i$, we construct a test FSA that checks whether or not $\mathcal{D}_i = \mathcal{D}_{\mathrm{Geo}}$. If the test passes, we halt, have verified hyperbolicity, and $\mathrm{FtGeo}_i = \mathrm{Geo}(X)$. Failure yields $\mathcal{D}_{i+1}$; we increment $i$ and loop.

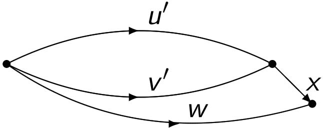
How do we construct the test FSA? The test should fail if for some $w \in L(\mathcal{W})$, $\exists v, u$ with $(u, v), (v, w) \in \mathcal{D}_i^{\epsilon}$, $(u, v) \in \mathcal{D}_i^{\epsilon}$, $|u| = |v| = |w|$, but $(u, w) \notin \mathcal{D}_i^{\epsilon}$.



# Building a test FSA

Such a test was provided in Wakefield's 1997 thesis, which contained a procedure based on the construction of an FSA $\mathcal{T}_i$ with language

$$\left\{ \begin{array}{lll} u' & : & \exists v', w \in L(\mathcal{W}), x \in X^{\pm}, |w| = |v'| + 1, \\ & & (v', w) \in L(\mathcal{D}^x_{\mathcal{W}}),\ (v', u') \in L(\mathcal{D}^{\epsilon}_i), (w, u'x) \notin L(\mathcal{D}^{\epsilon}_i) \end{array} \right\}$$




If $L(\mathcal{T}_i) = \emptyset$, then hyperbolicity is verified, $\mathcal{D}_i = \mathcal{D}_{\text{Geo}}$ and $\text{FtGeo}_i = \text{Geo}(X)$; otherwise $\mathcal{D}^{\epsilon}_{i+1}$ must accept $L(\mathcal{D}^{\epsilon}_i) \cup \{(u'x, w)\}$.

A later procedure, described by Epstein&Holt (1998) is more efficient, through its use of a two-string test FSA $\mathcal{T}'$ with language

$$\{(u, w) : w \in \mathcal{W}, \exists v, (u, v), (v, w) \in L(\mathcal{D}^{\epsilon}_i),\ |u| = |v| = |w|, (u, w) \notin \mathcal{D}^{\epsilon}_i\}$$

Hyperbolicity is verified if $L(\mathcal{T}'_i) = \emptyset$; otherwise new word differences are found.

# Conjugacy problem in hyperbolic groups

$O(n^2)$ and $O(n)$ solutions to $\mathtt{CP}(G)$ in hyperbolic $G$ are described by Bridson&Haefliger (1999) and Epstein&Holt (2006), but are impractical. A practical $O(n^3)$ solution restricting to inf. order elements due to Marshall (2008), uses ideas from Swenson, has been implemented in GAP. We reduce to the question of conjugacy between straight elements:

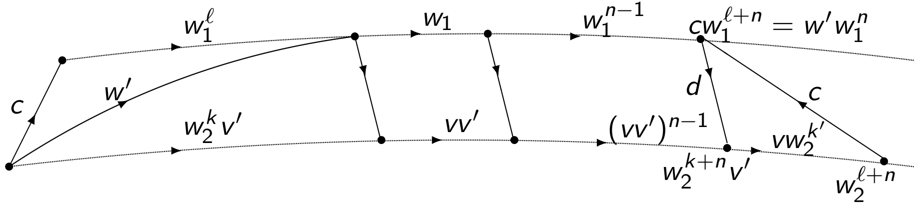
**Definition**

A word $w$ over $X$ is **straight** if all powers $w^n$ with $n \geq 0$ are geodesic. An element $g \in G$ is straight if represented by a straight word $w$.

If $g \in G$ has inf. order, then, $\exists h, m$, s.t. $hg^m h^{-1}$ is straight.

# Testing for conjugacy between $\infty$-order elts rep.by $u_1, u_2$.

Find $c_1, c_2, m$ st elts $(c_1 u_1 c_1^{-1})^m$ and $(c_2 u_2 c_2^{-1})^m$ are straight, rep by straight words $w_1, w_2$. if $u_1, u_2$ are conjugate, then $\exists c$, $cw_1^n =_G w_2^n c$ for all $n$. The following picture shows us that $c = dv$, where $d$ is a word difference between the two (fellow travelling) infinite rays and $w_2 = v'v$:




So we check all pairs $(d, v)$ st $d$ is a word difference of a particular difference machine $\mathcal{D}_\infty$ and $w_2 = v'v$ to see if $dv$ conjugates $w_1$ to $w_2$

For each pair $(d, v)$ for which $dv$ conjugates $w_1$ to $w_2$, check to see if $dv$ conjugates $c_1 u_1 c_1^{-1}$ to $c_2 u_2 c_2^{-1}$.

$u_1$ and $u_2$ are conjugate $\iff$ some such pair $(d, v)$ can be found.