Solving equations in groups

Laura Ciobanu

Heriot-Watt University (Edinburgh, UK)

Discrete Maths Seminar, Monash University

4 September 2019

EPSRC
Engineering and Physical Sciences
Research Council

Australian Government
Australian Research Council

Equations in monoids and groups - introduction

Equations in monoids: word equations

- Algebraic structures: monoids.

Free monoid $FM(a,b)$ = all words over $\{a,b\}$.

Equations in monoids: word equations

- Algebraic structures: monoids.

Free monoid $FM(a, b) =$ all words over $\{a, b\}$.

- An equation in the structure and variables $\{X, Y, \dots\}$ has the form

$$XaUZaU = YZbXaabY \text{ in } FM(a, b).$$

Equations in monoids: word equations

- Algebraic structures: monoids.

Free monoid $FM(a, b)$ = all words over $\{a, b\}$.

- An equation in the structure and variables $\{X, Y, \ldots\}$ has the form

$$XaUZaU = YZbXaabY \text{ in } FM(a, b).$$

- A *solution* is an assignment for each variable that makes the two sides equal.

$$X \to abb,\ Y \to ab,\ Z \to ba,\ U \to bab \implies$$

$$XaUZaU = YZbXaabY = abbababbaabab.$$

Equations in groups

- G is a group
- $\{X_1, \ldots, X_n\}$ is a set of variables.

An *equation* with coefficients g_j in G has the form

$$g_1 X_{i_1}^{\epsilon_1} g_2 X_{i_2}^{\epsilon_2} \ldots X_{i_m}^{\epsilon_m} g_{m+1} = 1_G$$

where $i_j \in \{1, \ldots, n\}, \epsilon_j \in \{1, -1\}$.

Equations in free groups

$FG(a, b)$ = the free group on a, b.

- The equation

$$X^{-1}abX = ba$$

Equations in free groups

$FG(a, b) =$ the free group on a, b.

- The equation

$$X^{-1}abX = ba$$

has solutions $X = (ab)^n a,\ n \in \mathbb{Z}$.

Equations in free groups

$FG(a, b) =$ the free group on a, b.

- The equation

$$X^{-1}abX = ba$$

has solutions $X = (ab)^n a$, $n \in \mathbb{Z}$.

- The equation

$$XYX^{-1}Y^{-1} = aba^{-1}b^{-1}$$

has solutions $X = a$, $Y = b$; $X = ab$, $Y = b$; $\dots$, $X = ab^n$, $Y = b$;

$X = a$, $Y = ba^m$ $\dots$

Two natural questions

- **Deciding satisfiability: a decision problem**

  Does an equation have solutions? Does there exist an algorithm which for any equation in a given group can determine whether the equation is satisfiable?

- **Find and describe the solutions**

  Give an algorithm that finds a solution (all solutions) for any satisfiable equation.

Motivation: Hilbert's Tenth Problem

(Markov, Hmelevskii, Malcev, Makanin, ...)

- The matrices $a = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ and $b = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$ form a free monoid inside $SL_2(\mathbb{Z})$.

- Consider a word equation over $\{a, b\}^*$ with variables $\{X_1, \ldots, X_n\}$, where each $X_i = \begin{pmatrix} \alpha_{i1} & \alpha_{i2} \\ \alpha_{i3} & \alpha_{i4} \end{pmatrix}$, $\alpha_{ij} \in \mathbb{N}$.

- Consider an equation over $\{a, b\}^*$ with variables $\{X_1, \ldots, X_n\}$, where each $X_i = \begin{pmatrix} \alpha_{i1} & \alpha_{i2} \\ \alpha_{i3} & \alpha_{i4} \end{pmatrix}$, $\alpha_{ij} \in \mathbb{Z}$.
- The equation becomes

$$\begin{pmatrix} P_1 & P_2 \\ P_3 & P_4 \end{pmatrix} = \begin{pmatrix} Q_1 & Q_2 \\ Q_3 & Q_4 \end{pmatrix},$$

  where $P_1, \ldots, Q_4$ are polynomials in the α_{ij}.
- The equation has a solution if and only if the Diophantine system has a solution:

$$\alpha_{i1}\alpha_{i4} - \alpha_{i2}\alpha_{i3} = 1$$

$$P_j = Q_j.$$

Deciding satisfiability of systems of equations is:

- Unsolvable in general groups/monoids.

Deciding satisfiability of systems of equations is:

- Unsolvable in general groups/monoids.
- First breakthrough: decidable, but hard over free (semi)groups (not primitive recursive, EXPSPACE)
  (G. Makanin 1982 and A. Razborov 1985)
- Decidable in groups that 'close' to free:

  - hyperbolic (Rips & Sela, 1995; Dahmani & Guirardel, 2010)

  - partially commutative (Diekert & Muscholl, 2005)

  - some extensions of the above

Equations in other groups

- Solvable in abelian groups: linear algebra

Equations in other groups

- Solvable in abelian groups: linear algebra
- Not solvable in general nilpotent groups

  (Roman'kov '79)

Equations in other groups

- Solvable in abelian groups: linear algebra
- Not solvable in general nilpotent groups

  (Roman'kov '79)
- Solvable for single equations in all Heisenberg groups

  (Duchin, Liang, Shapiro '14)

Equations that are not

An inequation with coefficients g_j in G has the form

$$g_1 X_{i_1}^{\epsilon_1} g_2 X_{i_2}^{\epsilon_2} \dots X_{i_m}^{\epsilon_m} g_{m+1} \neq 1_G$$

where $i_j \in \{1, \dots, n\}, \epsilon_j \in \{1, -1\}$.

Connections to:

- Logic: the satisfiability of equations and inequations is equivalent to the decidability of the existential theory of a group.

Connections to:

- Logic: the satisfiability of equations and inequations is equivalent to the decidability of the existential theory of a group.

  (Tarski's Conjecture $\sim$1950) The elementary theories of free groups with different number of generators coincide. (Sela, Kharlampovich & Myasnikov)

Connections to:

- Logic: the satisfiability of equations and inequations is equivalent to the decidability of the existential theory of a group.

  (Tarski's Conjecture $\sim$1950) The elementary theories of free groups with different number of generators coincide. (Sela, Kharlampovich & Myasnikov)

- Geometry: Viewing maps from surface groups to free groups geometrically $\Longrightarrow$ quadratic equations (every variable appears twice) in free groups are well understood.

Connections to:

- Logic: the satisfiability of equations and inequations is equivalent to the decidability of the existential theory of a group.

  (Tarski's Conjecture $\sim$1950) The elementary theories of free groups with different number of generators coincide. (Sela, Kharlampovich & Myasnikov)

- Geometry: Viewing maps from surface groups to free groups geometrically $\Longrightarrow$ quadratic equations (every variable appears twice) in free groups are well understood.

- Theoretical Computer Science

  Unification theory - solvability of free monoid equations.

Understanding the solutions

Description of solutions: algebraic approach

1987 Razborov: Description of all solutions for an equation in a free group via "Makanin-Razborov" diagrams.

Description of solutions: algebraic approach

1987 Razborov: Description of all solutions for an equation in a free group via "Makanin-Razborov" diagrams.

2000s: generalisations of such diagrams to hyperbolic groups etc

Description of solutions: algebraic approach

1987 Razborov: Description of all solutions for an equation in a free group via "Makanin-Razborov" diagrams.

2000s: generalisations of such diagrams to hyperbolic groups etc

2016 Sela: Description of all solutions for an equation in a free monoid via "Makanin-Razborov" diagrams.

Solving equations in free structures: the CS approach

- 1999 Plandowski: Decidability for word equations is in PSPACE.
- 2000 Gutiérrez: Decidability for free group equations is in PSPACE.
- 2001 Diekert, Gutiérrez, Hagenah: Decidability for free group equations with *rational constraints* is PSPACE-complete.
- 2013: Jeż applied *recompression* and simplified all known proofs for decidability.
- 2014: Diekert, Jeż, Plandowski gave a new PSPACE algorithm that produces all solutions for an equation with rational constraints in free groups or free monoids with involution.

Description of solutions as formal languages

Let A be a finite alphabet. A formal language over A is a set $L \subset A^*$ of words.

Description of solutions as formal languages

Let A be a finite alphabet. A formal language over A is a set $L \subset A^*$ of words.

How do we represent the solutions as a formal language?

Description of solutions as formal languages

Let A be a finite alphabet. A formal language over A is a set $L \subset A^*$ of words.

How do we represent the solutions as a formal language?

- **In** $FG(a,b)$: represent the solutions of $XYX^{-1}Y^{-1} = aba^{-1}b^{-1}$ as

$$\{a\#b, ab\#b, ab^2\#b, \dots\}$$

over the alphabet $\{a, b, a^{-1}, b^{-1}, \#\}$.

- **In** $FG(a,b)$: represent the solutions of $XYX^{-1}Y^{-1} = aba^{-1}b^{-1}$ as

$$\{a\#b, ab\#b, ab^2\#b, \dots\}$$

over the alphabet $\{a, b, a^{-1}, b^{-1}, \#\}$.

- **In** $FG(a,b)$: represent the solutions of $XYX^{-1}Y^{-1} = aba^{-1}b^{-1}$ as

$$\{a\#b, ab\#b, ab^2\#b, \dots\}$$

over the alphabet $\{a, b, a^{-1}, b^{-1}, \#\}$.

- **In general**: Let G be a group generated by a set A.
    - Suppose $U = 1$ is an equation over G with variables $\Omega = \{X_1, \dots, X_k\}$.

- **In** $FG(a,b)$: represent the solutions of $XYX^{-1}Y^{-1} = aba^{-1}b^{-1}$ as

$$\{a\#b, ab\#b, ab^2\#b, \dots\}$$

over the alphabet $\{a, b, a^{-1}, b^{-1}, \#\}$.

- **In general**: Let G be a group generated by a set A.
  - Suppose $U = 1$ is an equation over G with variables $\Omega = \{X_1, \dots, X_k\}$.
  - Any solution of $U = 1$ is a substitution σ with $\sigma(X_i) = u_i$, $u_i \in A^*$.
  - Let $\#$ be a symbol not in A. We encode a solution of $U = 1$ as

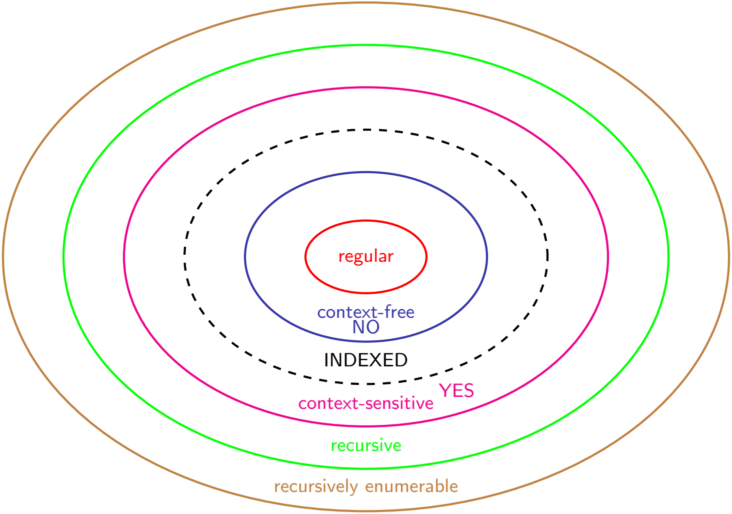
$$\sigma(X_1)\# \cdots \#\sigma(X_k).$$

- What is the formal language type of a solution set in a free group?

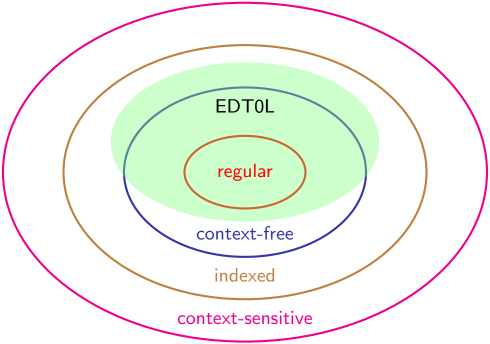
- What is the formal language type of a solution set in a free group?
- ANSWER: An indexed language!

  (C., Diekert, Elder)

The hierarchy of formal languages



The right kind of language: EDT0L $\subset$ indexed



Solutions are EDT0L in important classes of groups

CDE The set of all solutions as reduced words in a free group is EDT0L in NSPACE($n \log n$).

(C. - Diekert - Elder, ICALP 2015)

DE The set of all solutions in a virtually free group is EDT0L in NSPACE($n^2 \log n$).

(Diekert - Elder, ICALP 2017)

DJK The set of all solutions in a partially commutative group is EDT0L in NSPACE($n \log n$).

(Diekert - Jeż - Kufleitner, ICALP 2016)

EDT0L (**E**_xtended,_ **D**_eterministic,_ **T**_able,_ **0** _interaction, and_ **L**_indenmayer_)

- 1960s: Lindenmeyer's grammars intended to model the growth of organisms.

EDT0L (**E***xtended*, **D***eterministic*, **T***able*, **0** *interaction, and* **L***indenmayer*)

- 1960s: Lindenmeyer's grammars intended to model the growth of organisms.
- L languages: GREAT for ALGEBRA. Why?

EDT0L (**E***xtended,* **D***eterministic,* **T***able,* **0** *interaction, and* **L***indenmayer*)

- 1960s: Lindenmeyer's grammars intended to model the growth of organisms.
- L languages: GREAT for ALGEBRA. Why?

  Obtained by applying a (rational) family of morphisms* to some fixed symbol $\#$.

EDT0L (**E***xtended,* **D***eterministic,* **T***able,* **0** *interaction, and* **L***indenmayer*)

- 1960s: Lindenmeyer's grammars intended to model the growth of organisms.
- L languages: GREAT for ALGEBRA. Why?

  Obtained by applying a (rational) family of morphisms* to some fixed symbol $\#$.

- Definition:

  Let A be an alphabet and $L \subseteq A^*$.

  L is an EDT0L language if there is

  - an alphabet $C \supseteq A$,

EDT0L (**E***xtended,* **D***eterministic,* **T***able,* **0** *interaction, and* **L***indenmayer*)

- 1960s: Lindenmeyer's grammars intended to model the growth of organisms.
- L languages: GREAT for ALGEBRA. Why?

  Obtained by applying a (rational) family of morphisms* to some fixed symbol $\#$.

- Definition:

  Let A be an alphabet and $L \subseteq A^*$.

  L is an EDT0L language if there is

  - an alphabet $C \supseteq A$,
  - a finite set H of morphisms $C^* \to C^*$,

EDT0L (*Extended, Deterministic, Table, 0 interaction, and Lindenmayer*)

- 1960s: Lindenmeyer's grammars intended to model the growth of organisms.
- L languages: GREAT for ALGEBRA. Why?

  Obtained by applying a (rational) family of morphisms* to some fixed symbol $\#$.

- Definition:

  Let A be an alphabet and $L \subseteq A^*$.

  L is an EDT0L language if there is

  - an alphabet $C \supseteq A$,
  - a finite set H of morphisms $C^* \to C^*$,
  - a rational language $R \subseteq H^*$ of morphisms, and

EDT0L (**E**_xtended,_ **D**_eterministic,_ **T**_able,_ **0** _interaction, and_ **L**_indenmayer_)

- 1960s: Lindenmeyer's grammars intended to model the growth of organisms.
- L languages: GREAT for ALGEBRA. Why?

  Obtained by applying a (rational) family of morphisms* to some fixed symbol $\#$.

- Definition:

  Let A be an alphabet and $L \subseteq A^*$.

  L is an EDT0L language if there is

  - an alphabet $C \supseteq A$,
  - a finite set H of morphisms $C^* \to C^*$,
  - a rational language $R \subseteq H^*$ of morphisms, and
  - a symbol $\# \in C$ such that

$$L = \{\phi(\#) \mid \phi \in R\} \cap A^*.$$

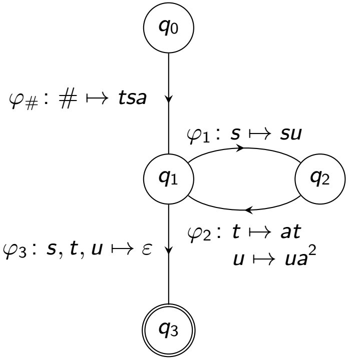
$L = \{a^{n^2} \mid n \in \mathbb{N}\}$ is EDT0L

$A = \{a\} \subset C = \{\#, s, t, u, a\}$

$L = \{a^{n^2} \mid n \in \mathbb{N}\}$ is EDT0L

$A = \{a\} \subset C = \{\#,\ s,\ t,\ u,\ a\}$

The finite state automaton $\mathcal{A}$ gives the rational control R:



Then $R = \varphi_\#(\varphi_1\varphi_2)^*\varphi_3$ applied to start symbol $\#$ gives $\{a^{n^2} \mid n \in \mathbb{N}\}$.

Theorem (C., Diekert, Elder, 2016)

Let $F(A)$ be the free group on A and $\Omega = \{X_1, \ldots, X_k\}$ a set of variables.

- The set of all solutions in reduced words to $U = V$ is an EDT0L language.
- There is an algorithm which takes (U, V) as input and computes in NSPACE($n \log n$) a finite graph (an NFA) $\mathcal{A}$ where the transitions are monoid morphisms and

$$Sol(U = V) = \{\phi(\$) \mid \phi \in L(\mathcal{A})\}.$$

The algorithm: an overview

1. First we translate an equation in a free group into a system of equations in a free monoid with involutions.

2. Then we solve equations in free monoids with RATIONAL constraints.

3. We ensure that the solutions are reduced words in the free group by using the rational constrains. (MR diagrams cannot produce reduced words!)

About the algorithm

1. It is easy to produce the graph that gives the solutions, HARD to show it is the correct graph.

2. Once the graph is produced from an initial vertex to final vertices, we start at the final vertices and go backwards to the initial ones to read off the solutions.

   This gives us the EDT0L description.

The proof: preprocessing

Step 1: transform equation into triangular system

Take the equation $U = V$ in $F(A)$, which is equivalent to $UV^{-1} = 1$,

and make a system of equations, using new variables Z_i, as follows:

$$\begin{aligned} UV^{-1} &= p_1 p_2 p_3 \dots p_n = 1 \\ &\to p_1 p_2 = Z_1,\ Z_1 p_3 = Z_2, \quad Z_2 p_4 = Z_3,\ Z_3 p_5 = Z_4, \dots \end{aligned}$$

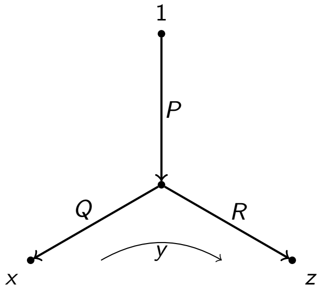
Each equation is now *triangular*, i.e. it has length 3.

Step 2. Free groups $\longrightarrow$ free monoids

In a free group, $xy = z$ holds if and only if there are reduced words P, Q, R with

$$x = PQ, y = Q^{-1}R, z = PR$$

as *word equations* in the free monoid over $A = \{a_1, a_1^{-1}, \ldots, a_d, a_d^{-1}\}$.



Step 3: Free monoids with involution

- Write a^{-1} as $\overline{a}$ and X^{-1} as $\overline{X}$.

  The map $a \mapsto \overline{a} : A \to A$ is an *involution*, i.e. $\overline{(\overline{a})} = a$ for all $a \in A$.

- We have a system of *word equations* $k_i l_i = m_i$ over $A \cup \Omega$

  (where $A = \{a_1, \overline{a_1}, \ldots, a_d, \overline{a_d}\}$ and Ω now includes $\overline{X}_i, Z_i, \overline{Z}_i, P, \overline{P}$, etc.)

  and we require that solutions do not contain $a\overline{a}$ or $\overline{a}a$.

- Finally put the system $k_i l_i = m_i$ into a single equation

$$k_1 l_1 \# k_2 l_2 \# \ldots \# k_s l_s = m_1 \# m_2 \# \ldots \# m_s$$

  and insist that the letter $\# \notin A$ does not appear in any solution.

Step 3: Free groups $\longrightarrow$ free monoids with constraints

Now let $A := \{a_1, \overline{a_1}, \ldots, a_d, \overline{a_d}, \#\}$.

How do we find solutions $\{X_i \to u_i\}$ to a *word equation* such that:

- $u_i \in A^*$ is not allowed to contain any subwords $a\bar{a}$,
- $u_i \in A^*$ is not allowed to use the letter $\#$?

We use CONSTRAINTS.

The proof: key process

Main idea of the proof: example $aXXb = YX$

- We start *guessing* the first and last letters of the variables, and substitute:

  $X \to aX$ $\qquad$ $aaXaXb = YaX$

Main idea of the proof: example $aXXb = YX$

- We start *guessing* the first and last letters of the variables, and substitute:

  $X \to aX \qquad aaXaXb = YaX$

  $X \to Xb \qquad aaXbaXbb = YaXb$

Main idea of the proof: example $aXXb = YX$

- We start *guessing* the first and last letters of the variables, and substitute:

  $X \to aX \qquad aaXaXb = YaX$

  $X \to Xb \qquad aaXbaXbb = YaXb$

  $X \to aX \qquad aaaXbaaXbb = YaaaXb$

- We get *long segments* of constants in between variables.
- We *compress* these segments, to bring the equation length back down.
- Eventually, we will substitute $X \to 1$ and $Y \to 1$, and be left with two words just in constants. If they are identical we accept, else reject.

Goal: Find all solutions to the word equation $U = V$ satisfying the constraints.

- We first *guess* $\rho(X) \in N \setminus \{0\}$ for each $X \in \Omega$.

- We then apply the following moves to the equation:

  - *pop* variables: $X \to aX$
  - *compress pairs* of constants $ab \to c$ where c is a new constant.
  - *compress blocks* of letters $aa \ldots a \to a_\ell$ where a_ℓ is a new constant.
  - Eventually, we will substitute $X \to 1$ and $Y \to 1$, and be left with two words just in constants. If they are identical we accept, else reject.

Comments

- The first move (pop) increases the length of the equation, but gets us closer to a solution.
- The two compression moves, applied many times, will reduce the length of the equation, at the expense of enlarging the set of constants.

Constructing the NFA: the vertices

We represent this process using a finite directed graph.

Vertices — labeled by the current state of the equation, plus some extra data.

- An initial vertex is a vertex containing the initial equation together with some guess for the constraint for each variable X.
- Final vertices — equation with no variables and both sides identical.

Constructing the NFA: the transitions

(I) As we move between vertices, variables will be replaced (*eg* $X \to aX$), and the value of ρ will be updated.

(II) Also, we have two types of compression:

- pair: $ab \to c_{ab}$
- block: $bbb \dots b \to b_\ell$

A priori these restrictions might mean that we miss finding some (any) solutions. The heart of the proof is to show that, with the right bounds, the graph encodes precisely all the solutions.

The nondeterministic finite automaton

So we need *more constants* than the original set A. Call this set C.

We define several types of edges, such that solutions and constraints are preserved by an edge, and each edge is labeled by a morphism h of C^*.

To ensure the graph is *finite*, we must

- only use (and reuse) finitely many new constants,
- have a GLOBAL BOUND on the length of any intermediate equation.

How do we get the solutions?

If there is a path from some

- initial vertex to a
- final vertex (with $P = P$ and no variables)

then we can apply the maps h labeling the path *backwards* from final to initial and recover a solution to $U = V$.

Is this graph the correct one?

The graph can then be turned into a finite state automaton, accepting a language R of morphisms. The set of all solutions becomes the (EDT0L) language $\{h(\$) \mid h \in R\}$.

The key of the proof is to show

(1) that every answer we get is indeed a solution, and

(2) we get all solutions.

Dealing with the two issues

(1) every answer we get is indeed a solution:

the graph was constructed to preserve solutions,

(2) we get all solutions:

the most technical and complicated part of the paper.

Solutions sets to systems of equations in

hyperbolic groups

Groups and their presentations

Every group has a presentation $\mathcal{P} = \langle X \mid R \rangle$, which is an abstract way of defining the group via generators X and relations R.

Examples:

- $\mathcal{P} = \langle a, b, c \mid aba = bab, bcb^{-1} = c^2 \rangle$

Groups and their presentations

Every group has a presentation $\mathcal{P} = \langle X \mid R \rangle$, which is an abstract way of defining the group via generators X and relations R.

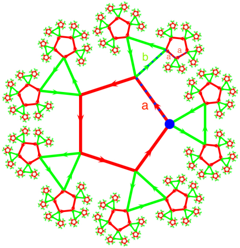
Examples:

- $\mathcal{P} = \langle a, b, c \mid aba = bab, bcb^{-1} = c^2 \rangle$
- $(\mathbb{Z}, +)$ has presentation $\langle a \mid \ \rangle$
- $(\mathbb{Z}^2, +)$ has presentation $\langle a, b \mid ab = ba \rangle$

Hyperbolic groups

Motivation: Most (finitely presented, i.e. X, R finite) groups are hyperbolic.

Definition: Groups whose 'picture' looks like the hyperbolic plane.



Examples: free groups, free products of finite groups, $SL(2,\mathbb{Z})$, virtually free groups *, surface groups, small cancellation groups, and many more.

---

* Virtually free = groups with a free subgroup of finite index.

Equations in hyperbolic groups

- Free products of finite groups:

$$G = \langle a, b \mid a^3 = 1, b^5 = 1 \rangle$$

Solving $XaYb^{-1} = Za^2X^{-1}$ means finding X, Y, Z in $\{a^{\pm 1}, b^{\pm 1}\}^*$, with

$$a^3 = 1, b^5 = 1.$$

Equations in hyperbolic groups

- Free products of finite groups:

$$G = \langle a, b \mid a^3 = 1, b^5 = 1 \rangle$$

Solving $XaYb^{-1} = Za^2X^{-1}$ means finding X, Y, Z in $\{a^{\pm 1}, b^{\pm 1}\}^*$, with

$$a^3 = 1, b^5 = 1.$$

- Surface groups:

$$G = \langle a, b, c, d \mid aba^{-1}b^{-1}cdc^{-1}d^{-1} = 1 \rangle$$

Solving $XaY^2b^{-1} = Ya^2X^{-1}$ means finding X, Y in $\{a^{\pm 1}, b^{\pm 1}, c^{\pm 1}, d^{\pm 1}\}^*$, with

$$aba^{-1}b^{-1}cdc^{-1}d^{-1} = 1.$$

Languages in groups

Let G be a group with finite symmetric generating set X. A language (over X) of

- normal forms picks one representative for each element in G.

Languages in groups

Let G be a group with finite symmetric generating set X. A language (over X) of

- normal forms picks one representative for each element in G.
  - shortlex normal forms picks the smallest representative for each element with respect to $<$, an order on X ($<_{sl}$ the induced shortlex order on X^*).

Languages in groups

Let G be a group with finite symmetric generating set X. A language (over X) of

- normal forms picks one representative for each element in G.
    - shortlex normal forms picks the smallest representative for each element with respect to $<$, an order on X ($<_{sl}$ the induced shortlex order on X^*).

Example:

Standard normal forms in $FG(a, b)$

$$= \text{the reduced words over } X = \{a^{\pm 1}, b^{\pm 1}\}$$

$$= X^* \setminus (X^* \cdot \{aa^{-1}, a^{-1}a, bb^{-1}, b^{-1}b\} \cdot X^*)$$

$\Longrightarrow$ the normal forms are a regular language!

Languages in groups

Let G be a group with finite symmetric generating set X. A language (over X) of

- normal forms picks one representative for each element in G.
  - shortlex normal forms picks the smallest representative for each element with respect to $<$, an order on X ($<_{sl}$ the induced shortlex order on X^*).

Example:

Standard normal forms in $FG(a, b)$

$$= \text{the reduced words over } X = \{a^{\pm 1}, b^{\pm 1}\}$$

$$= X^* \setminus (X^* \cdot \{aa^{-1}, a^{-1}a, bb^{-1}, b^{-1}b\} \cdot X^*)$$

$\Longrightarrow$ the normal forms are a regular language!

The set of shortlex normal forms is regular in hyperbolic groups.

Main Result (C. - Elder, 2019)

Let G be a hyperbolic group with generating set X.

- The set of all solutions in shortlex normal form to an equation $U = 1$ is an EDT0L language over the alphabet X.

Main Result (C. - Elder, 2019)

Let G be a hyperbolic group with generating set X.

- The set of all solutions in shortlex normal form to an equation $U = 1$ is an EDT0L language over the alphabet X.
- The complexity of building the graph which gives the EDT0L description is NSPACE($n^2 \log n$) if G is torsion-free*; otherwise it is NSPACE($n^4 \log n$), where n is the size of the equation.

*Torsion-free means $g^k \neq 1$ for all nontrivial $g \in G$ and nonzero integer k.

Applications

1. The *existential theory* for hyperbolic groups can be decided in NSPACE($n^2 \log n$) for torsion-free and NSPACE($n^4 \log n$) for groups with torsion.

Applications

1. The *existential theory* for hyperbolic groups can be decided in NSPACE($n^2 \log n$) for torsion-free and NSPACE($n^4 \log n$) for groups with torsion.

2. Can decide in the same space complexity as above whether or not the *solution set is empty, finite or infinite*.

Proof – big picture:

- **Torsion-free case** (Rips & Sela): solutions in G can be deduced from solving equations in $F(X)$.

Proof – big picture:

- **Torsion-free case** (Rips & Sela): solutions in G can be deduced from solving equations in $F(X)$.

  Use the CDE algorithm to get solutions in $F(X)$.

Proof – big picture:

- **Torsion-free case** (Rips & Sela): solutions in G can be deduced from solving equations in $F(X)$.

  Use the CDE algorithm to get solutions in $F(X)$.

- **Torsion/general case** (Dahmani & Guirardel): solutions in G are projections of solutions of equations in a virtually free group $V \twoheadrightarrow G$.

  Use the DE algorithm to get solutions in V.

To get EDT0L solutions: use above descriptions, plus the geometry of hyperbolic groups, plus language theory operations and results.

Thank you!