Working with Quotients of Finitely Presented Groups

Alexander Hulpke

Department of Mathematics

The Ohio State University

Columbus, OH

ahulpke@math.ohio-state.edu

http://www.math.ohio-state.edu/~ahulpke

From Summer 2001:

Department of Mathematics

Colorado State University

Fort Collins, CO

Two ways of specifying a subspace of R^n:

- By generators (spanning vectors).

Two ways of specifying a subspace of R^n:

- By generators (spanning vectors).
- By relations (nullspace of a matrix).

Two ways of specifying a subspace of R^n:

- By generators (spanning vectors).
- By relations (nullspace of a matrix).

Often problems specify relations and expect a solution in terms of a generating system.

Two ways of specifying a subspace of R^n:

- By generators (spanning vectors).
- By relations (nullspace of a matrix).

Often problems specify relations and expect a solution in terms of a generating system.

Analoguous for groups:

Vector space	$\longrightarrow$	Free group
Matrix rows	$\longrightarrow$	Relations
Nullspace	$\longrightarrow$	Factor group
Basis	$\longrightarrow$	Faithful representation

Finitely presented groups

A group given this way is a *finitely presented group*, given by a presentation

$$\langle \underline{\mathbf{x}} \mid \mathcal{R} \rangle = \langle x_1, \ldots, x_n \mid l_1(\underline{\mathbf{x}}) = r_1(\underline{\mathbf{x}}), \ldots, l_m(\underline{\mathbf{x}}) = r_m(\underline{\mathbf{x}}) \rangle.$$

$$\langle x, y \mid x^2 = 1, y^2 = 1, xyx = yxy \rangle$$

Finitely presented groups

A group given this way is a *finitely presented group*, given by a presentation

$$\langle \underline{\mathbf{x}} \mid \mathcal{R} \rangle = \langle x_1, \ldots, x_n \mid l_1(\underline{\mathbf{x}}) = r_1(\underline{\mathbf{x}}), \ldots, l_m(\underline{\mathbf{x}}) = r_m(\underline{\mathbf{x}}) \rangle.$$

$$\langle x, y \mid x^2 = 1, y^2 = 1, xyx = yxy \rangle$$

If we (formally) consider objects of the form $\langle \cdot \mid \cdot \rangle$ we talk of a *Presentation*.

For Example

Presentations arise in topology: We can write down a presentation for the fundamental group of a simplicial complex with

- Each edge e gives a generator g_e.
- For each edge e in a (chosen) *spanning tree*, a relation $g_e = 1$.
- For each triangle with edges a,b,c, a relation $g_a g_b g_c = 1$

For Example

Presentations arise in topology: We can write down a presentation for the fundamental group of a simplicial complex with

- Each edge e gives a generator g_e.
- For each edge e in a (chosen) *spanning tree*, a relation $g_e = 1$.
- For each triangle with edges a,b,c, a relation $g_a g_b g_c = 1$

There are (obvious) simplifications (Tietze transformations).

For Example

Presentations arise in topology: We can write down a presentation for the fundamental group of a simplicial complex with

- Each edge e gives a generator g_e.
- For each edge e in a (chosen) *spanning tree*, a relation $g_e = 1$.
- For each triangle with edges a,b,c, a relation $g_a g_b g_c = 1$

There are (obvious) simplifications (Tietze transformations).

The group of a *knot* is the fundamental group of its complement

One can write down a presentation for it from a picture of the knot.

For Example

Presentations arise in topology: We can write down a presentation for the fundamental group of a simplicial complex with

- Each edge e gives a generator g_e.
- For each edge e in a (chosen) *spanning tree*, a relation $g_e = 1$.
- For each triangle with edges a,b,c, a relation $g_a g_b g_c = 1$

There are (obvious) simplifications (Tietze transformations).

The group of a *knot* is the fundamental group of its complement

One can write down a presentation for it from a picture of the knot. Two knots are equivalent, if their groups are isomorphic.

Arithmetic

$$\langle x, y \mid x^2 = 1, y^2 = 1, xyx = yxy \rangle$$

Elements are represented by words, but different words might give the same element: x and x^3.

Arithmetic

$$\langle x, y \mid x^2 = 1, y^2 = 1, xyx = yxy \rangle$$

Elements are represented by words, but different words might give the same element: x and x^3.

We can use the relations to simplify words ("collection"):

$$\underline{xx}yxyx = y\underline{xyx} = \underline{yy}xy = xy$$

Arithmetic

$$\langle x, y \mid x^2 = 1, y^2 = 1, xyx = yxy \rangle$$

Elements are represented by words, but different words might give the same element: x and x^3.

We can use the relations to simplify words ("collection"):

$$\underline{xx}yxyx = y\underline{xyx} = \underline{yy}xy = xy$$

Analogon: Division of a polynomials by an ideal in $k[x_1, \ldots, x_n]$, Gröbner Bases

Technical requirements

Words must get “shorter” so that the process will finish:

Technical requirements

Words must get “shorter” so that the process will finish:

- Alphabet, Words

Technical requirements

Words must get “shorter” so that the process will finish:

- Alphabet, Words
- Monoid of Words (add inverse generators for group)

Technical requirements

Words must get “shorter” so that the process will finish:

- Alphabet, Words
- Monoid of Words (add inverse generators for group)
- Admissible (invariant under translation) ordering on words

Technical requirements

Words must get "shorter" so that the process will finish:

- Alphabet, Words
- Monoid of Words (add inverse generators for group)
- Admissible (invariant under translation) ordering on words
- Relations make words shorter (presentation is *rewriting system*)

Technical requirements

Words must get "shorter" so that the process will finish:

- Alphabet, Words
- Monoid of Words (add inverse generators for group)
- Admissible (invariant under translation) ordering on words
- Relations make words shorter (presentation is *rewriting system*)

Examples of permissible orders

- Length, then lexicographic
- *Not* pure lexicographic

Technical requirements

Words must get "shorter" so that the process will finish:

- Alphabet, Words
- Monoid of Words (add inverse generators for group)
- Admissible (invariant under translation) ordering on words
- Relations make words shorter (presentation is *rewriting system*)

Examples of permissible orders

- Length, then lexicographic
- *Not* pure lexicographic
- Wreath product ordering $a_1 b_1 a_2 b_2 \cdots a_k b_k$:
  First compare a-parts, if equal compare b-parts.

Problem

Potential ambiguity when multiple replacements are possible:

$$\underline{xx}yxyx = y\underline{xyx} = \underline{yy}xy = xy$$

$$x\underline{xyx}yx = xyx\underline{yy}x = xy\underline{xx} = xy$$

Here we were lucky

Problem

Potential ambiguity when multiple replacements are possible:

$$\underline{xx}yxyx = y\underline{xyx} = \underline{yy}xy = xy$$

$$x\underline{xyx}yx = xyx\underline{yy}x = xy\underline{xx} = xy$$

Here we were lucky

But in general we are not

$$\langle x, y \mid x^2 = 1, y^3 = 1, xyy = yxy \rangle$$

$$\underline{xyy}y = yxyy, \qquad x\underline{yyy} = x$$

Confluence

If *every* word eventually reduces uniquely, a rewriting system is called *confluent*.

One can then use the normal form to compute with elements.

Confluence

If *every* word eventually reduces uniquely, a rewriting system is called *confluent*.

One can then use the normal form to compute with elements.

Criterion: All overlaps of left hand sided of rules must reduce to the same word.

Confluence

If *every* word eventually reduces uniquely, a rewriting system is called *confluent*.

One can then use the normal form to compute with elements.

Criterion: All overlaps of left hand sided of rules must reduce to the same word.

Overlap $xx \to 1$ and $xyx \to yxy$:

$$\begin{aligned} \underline{xx}yx &= yx \\ x\underline{xyx} &= \underline{xyx}y = yx\underline{yy} = yx \end{aligned}$$

Knuth-Bendix "Algorithm"

(for obtaining a confluent rewriting system)

- Start with any rewriting system.

Knuth-Bendix "Algorithm"

(for obtaining a confluent rewriting system)

- Start with any rewriting system.
- Loop over all pairs of rules.

Knuth-Bendix "Algorithm"

(for obtaining a confluent rewriting system)

- Start with any rewriting system.
- Loop over all pairs of rules.
- Reduce each overlap of the left hand sides in both possible ways.

Knuth-Bendix "Algorithm"

(for obtaining a confluent rewriting system)

- Start with any rewriting system.
- Loop over all pairs of rules.
- Reduce each overlap of the left hand sides in both possible ways.
- If different normal forms arise, add a new rule to reduce the bigger one to the smaller.

Knuth-Bendix "Algorithm"

(for obtaining a confluent rewriting system)

- Start with any rewriting system.
- Loop over all pairs of rules.
- Reduce each overlap of the left hand sides in both possible ways.
- If different normal forms arise, add a new rule to reduce the bigger one to the smaller.
- (Rinse, Lather,) Repeat

Alas

The Knuth-Bendix procedure might not terminate.

This is not a problem of this algorithm but a fundamental difficulty:

Alas

The Knuth-Bendix procedure might not terminate.

This is not a problem of this algorithm but a fundamental difficulty:

> **Theorem:** (Boone, Novikov): *There cannot be an algorithm that will (dis)prove element equality in finitely presented groups (or finiteness of such a group)*

Alas

The Knuth-Bendix procedure might not terminate.

This is not a problem of this algorithm but a fundamental difficulty:

> **Theorem:** (Boone, Novikov): *There cannot be an algorithm that will (dis)prove element equality in finitely presented groups (or finiteness of such a group)*

Proof: Translation to Turing machines. Halteproblem.

Alas

The Knuth-Bendix procedure might not terminate.

This is not a problem of this algorithm but a fundamental difficulty:

> **Theorem:** (Boone, Novikov): *There cannot be an algorithm that will (dis)prove element equality in finitely presented groups (or finiteness of such a group)*

Proof: Translation to Turing machines. Halteproblem.

If G is finite, the method will terminate, but the runtime cannot be predicted.

Quotients

Therefore a principal tool is to study homomorphic images.

It is easy to check whether a mapping given on the generators is a homomorphism.

$$\langle a, b \mid a^2 = b^3 = (ab)^5 = 1 \rangle \to A_5, \quad a \mapsto (1,2)(3,4), b \mapsto (1,4,5)$$

Quotients

Therefore a principal tool is to study homomorphic images.

It is easy to check whether a mapping given on the generators is a homomorphism.

$$\langle a, b \mid a^2 = b^3 = (ab)^5 = 1 \rangle \to A_5, \quad a \mapsto (1,2)(3,4), b \mapsto (1,4,5)$$

Problems:

- How to find such a mapping?

Quotients

Therefore a principal tool is to study homomorphic images.

It is easy to check whether a mapping given on the generators is a homomorphism.

$$\langle a, b \mid a^2 = b^3 = (ab)^5 = 1 \rangle \to A_5, \quad a \mapsto (1,2)(3,4), b \mapsto (1,4,5)$$

Problems:

- How to find such a mapping?
- How to find a mapping with small kernel?

Quotients

Therefore a principal tool is to study homomorphic images.

It is easy to check whether a mapping given on the generators is a homomorphism.

$$\langle a, b \mid a^2 = b^3 = (ab)^5 = 1 \rangle \to A_5, \quad a \mapsto (1,2)(3,4), b \mapsto (1,4,5)$$

Problems:

- How to find such a mapping?
- How to find a mapping with small kernel?
- Given a φ, can we find a *lift* (i.e. ψ with $\ker \psi < \ker \varphi$)?

What homomorphisms can we get?

Abelian quotients

We get the largest abelian quotient by abelianizing the presentation.

The Smith normal form of this presentation gives the normal form as an abelian group:

$$\langle a, b \mid a^3 = b^2 = ababab = 1 \rangle$$

Abelian quotients

We get the largest abelian quotient by abelianizing the presentation.

The Smith normal form of this presentation gives the normal form as an abelian group:

$$\langle a, b \mid a^3 = b^2 = ababab = 1 \rangle \to \begin{pmatrix} 3 & 0 \\ 0 & 2 \\ 3 & 3 \end{pmatrix}$$

Abelian quotients

We get the largest abelian quotient by abelianizing the presentation.

The Smith normal form of this presentation gives the normal form as an abelian group:

$$\langle a, b \mid a^3 = b^2 = ababab = 1 \rangle \to \begin{pmatrix} 3 & 0 \\ 0 & 2 \\ 3 & 3 \end{pmatrix} \to$$

$$\begin{pmatrix} 1 & 0 \\ 0 & 3 \\ 0 & 0 \end{pmatrix}$$

Abelian quotients

We get the largest abelian quotient by abelianizing the presentation.

The Smith normal form of this presentation gives the normal form as an abelian group:

$$\langle a, b \mid a^3 = b^2 = ababab = 1 \rangle \to \begin{pmatrix} 3 & 0 \\ 0 & 2 \\ 3 & 3 \end{pmatrix} \to$$

$$\begin{pmatrix} 1 & 0 \\ 0 & 3 \\ 0 & 0 \end{pmatrix} \to C_1 \times C_3 \cong C_3$$

Bigger groups

Every group can be considered to be formed as an iterated *extension* of simple groups, its *composition factors*.

Bigger groups

Every group can be considered to be formed as an iterated *extension* of simple groups, its *composition factors*.

$$GL_4(3) = \left\{ \begin{array}{l} C_2 \\ | \\ \quad PSL_4(3) \\ \quad | \\ \quad C_2 \end{array} \right\} = SL_4(3)$$

If all simple factors are cyclic, the group is called *solvable*.

Chief series

One way to get factors is to take a chain of normal subgroups and refine. Then the subfactors are direct products of simple groups: $3^5.2^4.2^3.(A_5)^{11}.M_{11}$

Chief series

One way to get factors is to take a chain of normal subgroups and refine. Then the subfactors are direct products of simple groups: $3^5.2^4.2^3.(A_5)^{11}.M_{11}$

The factors which are direct products of cyclic groups can be considered as vector spaces.

Chief series

One way to get factors is to take a chain of normal subgroups and refine. Then the subfactors are direct products of simple groups: $3^5.2^4.2^3.(A_5)^{11}.M_{11}$

The factors which are direct products of cyclic groups can be considered as vector spaces.

Paradigm: If there is a normal subgroup that is a vector space, solve the problem for the factor group and use linear algebra to lift the result to the whole group.

In the above example:

$$(A_5)^{11}.M_{11} \to 2^3.(A_5)^{11}.M_{11} \to 2^4.2^3.(A_5)^{11}.M_{11} \to 3^5.2^4.2^3.(A_5)^{11}.M_{11}$$

Chief series

One way to get factors is to take a chain of normal subgroups and refine. Then the subfactors are direct products of simple groups: $3^5.2^4.2^3.(A_5)^{11}.M_{11}$

The factors which are direct products of cyclic groups can be considered as vector spaces.

Paradigm: If there is a normal subgroup that is a vector space, solve the problem for the factor group and use linear algebra to lift the result to the whole group.

In the above example:

$$(A_5)^{11}.M_{11} \to 2^3.(A_5)^{11}.M_{11} \to 2^4.2^3.(A_5)^{11}.M_{11} \to 3^5.2^4.2^3.(A_5)^{11}.M_{11}$$

"Typically" the factor of the solvable radical is small.

Quotient Algorithms

There are algorithms that find (the largest)

- Abelian quotient

Quotient Algorithms

There are algorithms that find (the largest)

- Abelian quotient
- p-group quotient.

Quotient Algorithms

There are algorithms that find (the largest)

- Abelian quotient
- p-group quotient.
- solvable group quotient.

(Description later)

Permutation quotients

If $U \leq G$ then G acts on the set of right cosets of U in G by right multiplication:

$$Ug \cdot h = U(gh)$$

Permutation quotients

If $U \leq G$ then G acts on the set of right cosets of U in G by right multiplication:

$$Ug \cdot h = U(gh)$$

If we number the cosets, we get a transitive permutation representation $G\varphi$ with $U\varphi = \mathrm{Stab}_{G\varphi}(1)$.

Every transitive permutation representation is obtained this way.

Todd-Coxeter Method

If we have generators for U, we can compute this permutation representation:

Define cosets C_i step by step by generator images:
$C_1 = U$, $C_i \cdot g = C_j$, &c.

Todd-Coxeter Method

If we have generators for U, we can compute this permutation representation:

Define cosets C_i step by step by generator images:
$C_1 = U$, $C_i \cdot g = C_j$, &c.

Relators and subgroup generators let cosets coincide:
$Uaba = U$ if aba a generator of U; $Uabbb = Ua$ if $b^3 = 1$.

Todd-Coxeter Method

If we have generators for U, we can compute this permutation representation:

Define cosets C_i step by step by generator images:
$C_1 = U$, $C_i \cdot g = C_j$, &c.

Relators and subgroup generators let cosets coincide:
$Uaba = U$ if aba a generator of U; $Uabbb = Ua$ if $b^3 = 1$.

If the index $[G{:}U]$ is finite, the method will terminate

Todd-Coxeter Method

If we have generators for U, we can compute this permutation representation:

Define cosets C_i step by step by generator images:
$C_1 = U$, $C_i \cdot g = C_j$, &c.

Relators and subgroup generators let cosets coincide:
$Uaba = U$ if aba a generator of U; $Uabbb = Ua$ if $b^3 = 1$.

If the index $[G{:}U]$ is finite, the method will terminate *but one cannot predict the runtime*

(Otherwise let $U = \langle\rangle$ and we would have a check whether $|G| = 1$.)

Variant 1: Low Index/GQuotient

In a backtrack search: find all sets of permutations of a given degree — or from a specified image group — that fulfill the relations.

Variant 1: Low Index/GQuotient

In a backtrack search: find all sets of permutations of a given degree — or from a specified image group — that fulfill the relations.

This finds all transitive permutation representations of limited degree — and thus all subgroups of limited index (Low Index),

or all images of G isomorphic to a specific group (GQuotient).

Variant 1: Low Index/GQuotient

In a backtrack search: find all sets of permutations of a given degree — or from a specified image group — that fulfill the relations.

This finds all transitive permutation representations of limited degree — and thus all subgroups of limited index (Low Index),

or all images of G isomorphic to a specific group (GQuotient).

To reduce the search space consider images only up to conjugation (renumbering of points):

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ \downarrow & & & \downarrow \\ 1 & 4 & 5 & 2 \end{pmatrix} : \quad \begin{array}{lcl} a & \mapsto & (1,2,3,4) \\ b & \mapsto & (1,2) \end{array}, \quad \begin{array}{lcl} a & \mapsto & (1,4,5,2) \\ b & \mapsto & (1,4) \end{array}$$

Variant 2: Subgroup presentations

How can we find quotients of a subgroup:

Let $U \leq G$ (of finite index) and φ the permutation action on the cosets of U (so $U = \mathrm{Stab}(1)$).

Variant 2: Subgroup presentations

How can we find quotients of a subgroup:

Let $U \leq G$ (of finite index) and φ the permutation action on the cosets of U (so $U = \mathrm{Stab}(1)$). Assume that

$$\begin{array}{lll} r_i & & 1 \to i \\ g & \text{maps (via } \varphi\text{)} & i \to j \\ r_j & & 1 \to j \end{array}$$

Then $r_i \cdot g \cdot r_j^{-1}$ maps $1 \to i \to j \to 1$.

Variant 2: Subgroup presentations

How can we find quotients of a subgroup:

Let $U \leq G$ (of finite index) and φ the permutation action on the cosets of U (so $U = \mathrm{Stab}(1)$). Assume that

$$\begin{array}{lll} r_i & & 1 \to i \\ g & \text{maps (via } \varphi) & i \to j \\ r_j & & 1 \to j \end{array}$$

Then $r_i \cdot g \cdot r_j^{-1}$ maps $1 \to i \to j \to 1$. So $r_i g / r_j \in U$.

Theorem: (Schreier): U is generated by the elements $r_i g / r_j$ for all i and all generators g of G ($j = i^g$).
One can rewrite the presentation of G to a presentation of U in these generators.

Problem

The number of generators grows with the index of U.

Even with simplification (*Tietze transformations*) we cannot go too deep into G,

Problem

The number of generators grows with the index of U.

Even with simplification (*Tietze transformations*) we cannot go too deep into G,

in particular not to the kernel $\ker \varphi$ of a homomorphism.

Application: Quotient subgroups

On the computer we represent every subgroup $U \leq G$ by a pair $(\varphi, U\varphi)$ where $\ker \varphi \leq U$.

(Previously: "Coset Table".)

Advantages

- More effective storage for big index.
- Easier calculations: Calculate in the image of the homomorphism.
- Taking preimages becomes trivial.

Application: Quotient subgroups

On the computer we represent every subgroup $U \leq G$ by a pair $(\varphi, U\varphi)$ where $\ker \varphi \leq U$.

(Previously: "Coset Table".)

Advantages

- More effective storage for big index.
- Easier calculations: Calculate in the image of the homomorphism.
- Taking preimages becomes trivial.

However: We need methods to compute in the image group (for example a permutation group)

Calculationg with two subgroups

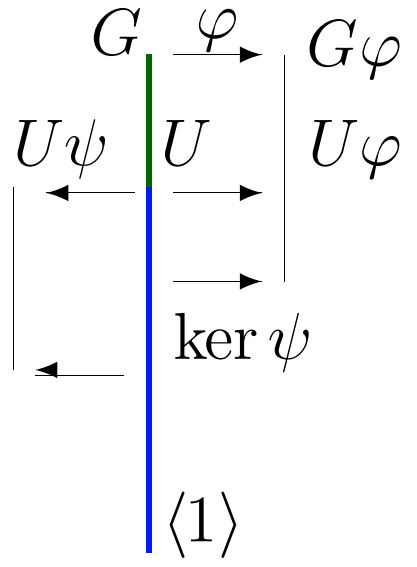
If there are two homomorphisms φ, ψ, we can map in the direct product $(G\varphi) \times (G\psi)$.

$$\varphi\colon \begin{cases} a & \mapsto & (1,2,3) \\ b & \mapsto & (1,2) \end{cases}, \psi\colon \begin{cases} a & \mapsto & (1,2) \\ b & \mapsto & (1,4,3,2) \end{cases}$$

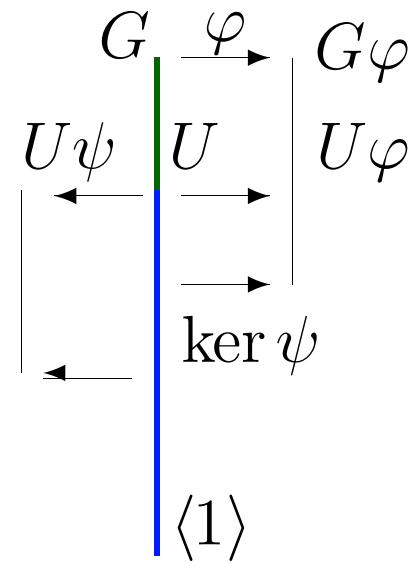
$$(\varphi,\psi)\colon \begin{cases} a & \mapsto & (1,2,3)(4,5) \\ b & \mapsto & (1,2)(4,7,6,5) \end{cases}$$

Induction

By rewriting, we can get a subgroup $V < U < G$ in quotient representation $(\psi, V\psi)$ related to U. How to get it related to G?



Induction



By rewriting, we can get a subgroup $V < U < G$ in quotient representation $(\psi, V\psi)$ related to U. How to get it related to G?

We can *induce* ψ to a homomorphism λ defined on G: (Permutation group version – "KRASNER-KALOUJNINE" theorem)

- Let φ the action of G on the $n = [G{:}U]$ cosets of U.

- Let φ the action of G on the $n = [G{:}U]$ cosets of U.
- $G\lambda$ permutes (via φ) n copies of the domain of $U\psi$ ("blocks"): $(1, \ldots, m), (m+1, \ldots, 2*m), \ldots, ((n-1)m+1, \ldots, n \cdot m)$

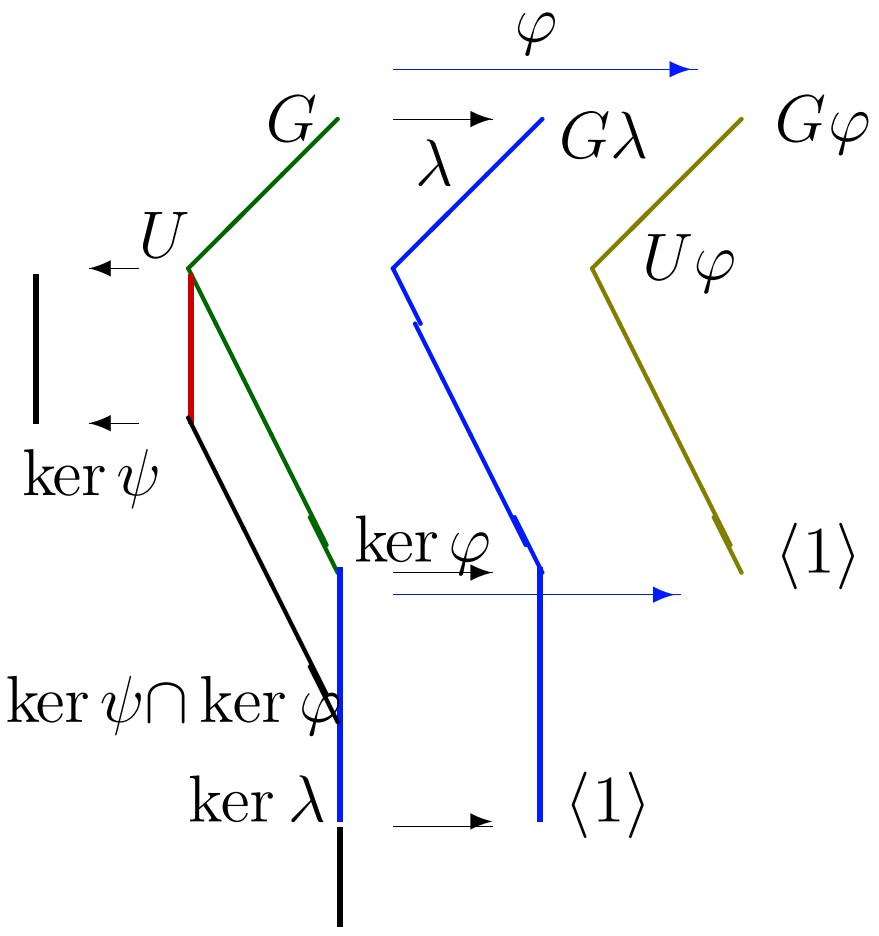
- Let φ the action of G on the $n = [G{:}U]$ cosets of U.
- $G\lambda$ permutes (via φ) n copies of the domain of $U\psi$ ("blocks"): $(1, \ldots, m), (m+1, \ldots, 2*m), \ldots, ((n-1)m+1, \ldots, n \cdot m)$
- $g \in G$ also acts on the i-th block as $(r_i g / r_j)\psi$.

- Let φ the action of G on the $n = [G{:}U]$ cosets of U.
- $G\lambda$ permutes (via φ) n copies of the domain of $U\psi$ ("blocks"): $(1, \ldots, m), (m+1, \ldots, 2*m), \ldots, ((n-1)m+1, \ldots, n \cdot m)$
- $g \in G$ also acts on the i-th block as $(r_i g / r_j)\psi$.
- λ is a *lift* of φ: $\ker \lambda \leq \ker \varphi$

- Let φ the action of G on the $n = [G{:}U]$ cosets of U.
- $G\lambda$ permutes (via φ) n copies of the domain of $U\psi$ ("blocks"): $(1, \ldots, m), (m+1, \ldots, 2*m), \ldots, ((n-1)m+1, \ldots, n \cdot m)$
- $g \in G$ also acts on the i-th block as $(r_i g / r_j)\psi$.
- λ is a *lift* of φ: $\ker \lambda \leq \ker \varphi$

If $V = Stab_U(1)$ (via ψ), then $V = \mathrm{Stab}_U(1)$ (via λ).

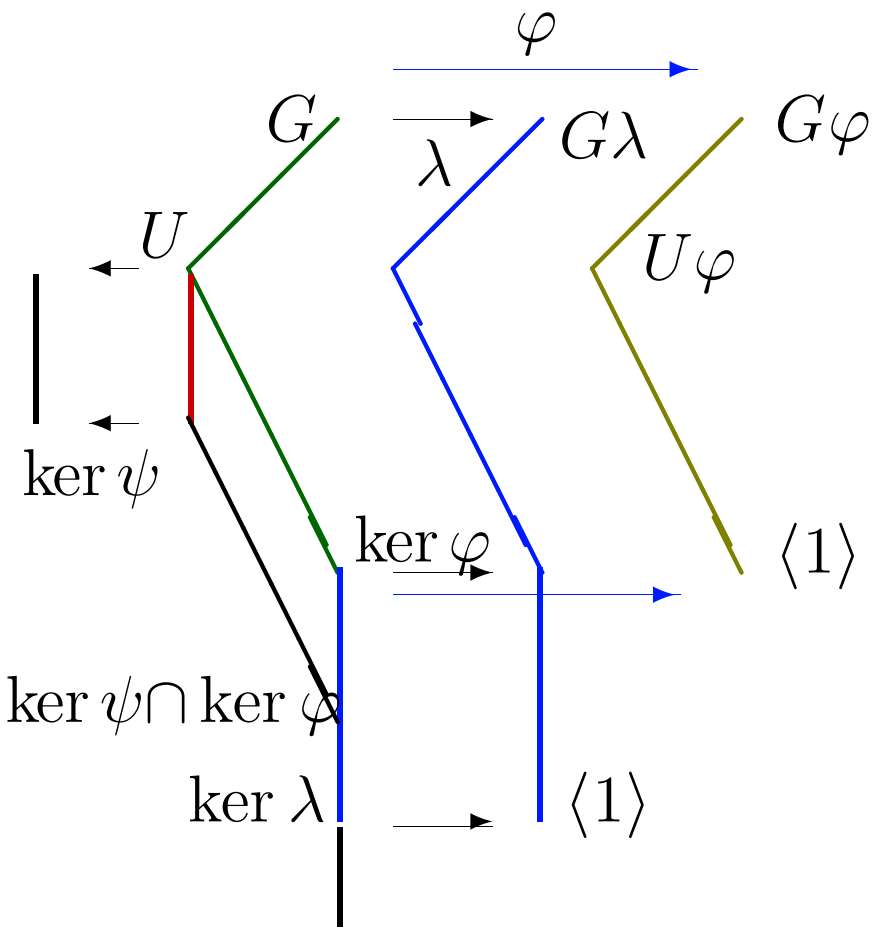
Finding new Quotients



Given G and φ:

- Take a subgroup U of $G\varphi$.

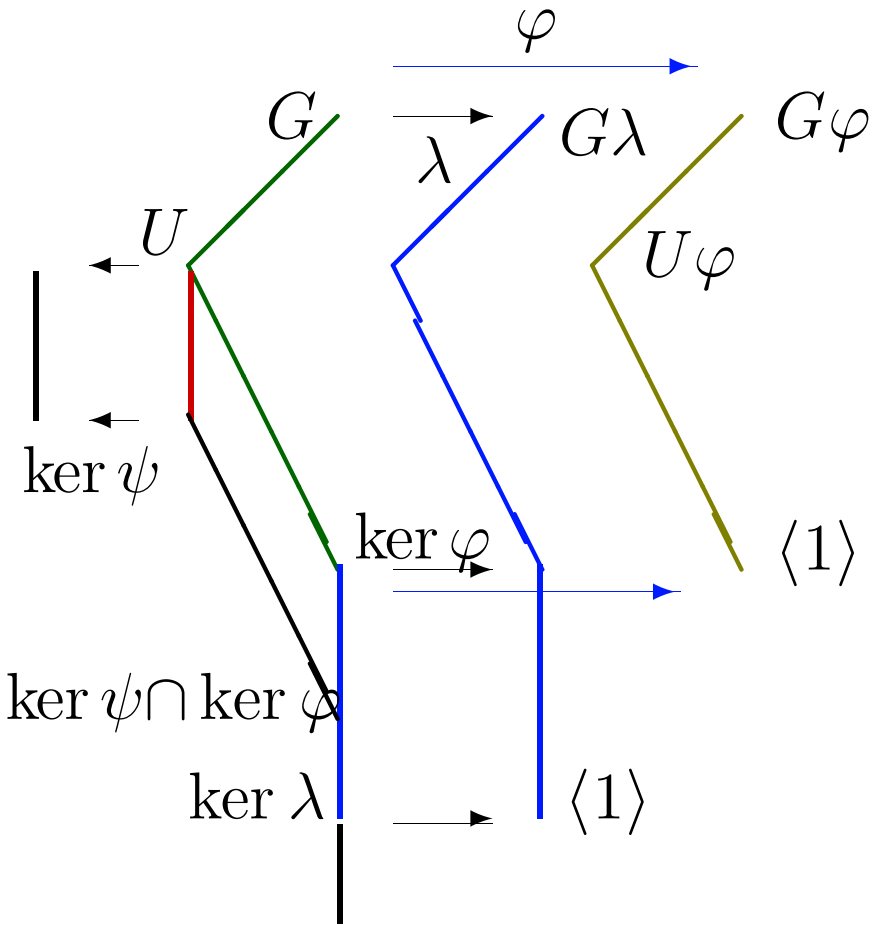
Finding new Quotients



Given G and φ:

- Take a subgroup U of $G\varphi$.
- Rewrite the presentation (Consider U as another finitely presented group)

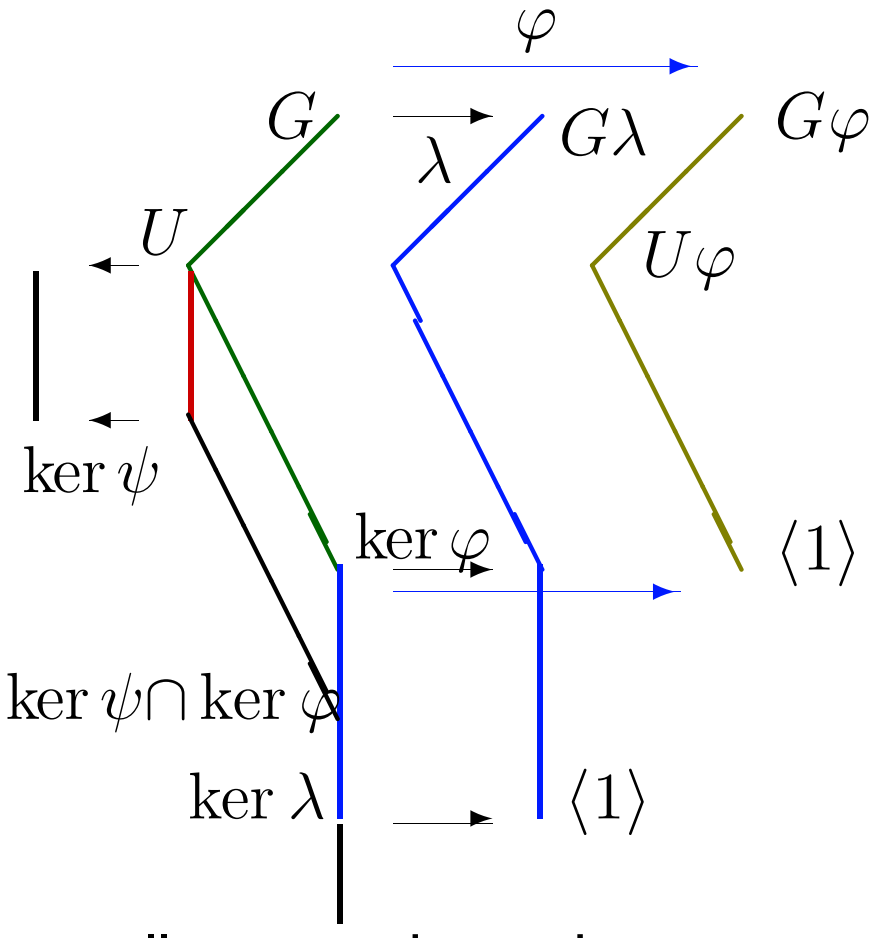
Finding new Quotients



Given G and φ:

- Take a subgroup U of $G\varphi$.
- Rewrite the presentation (Consider U as another finitely presented group)
- Find quotients ψ of U.

Finding new Quotients

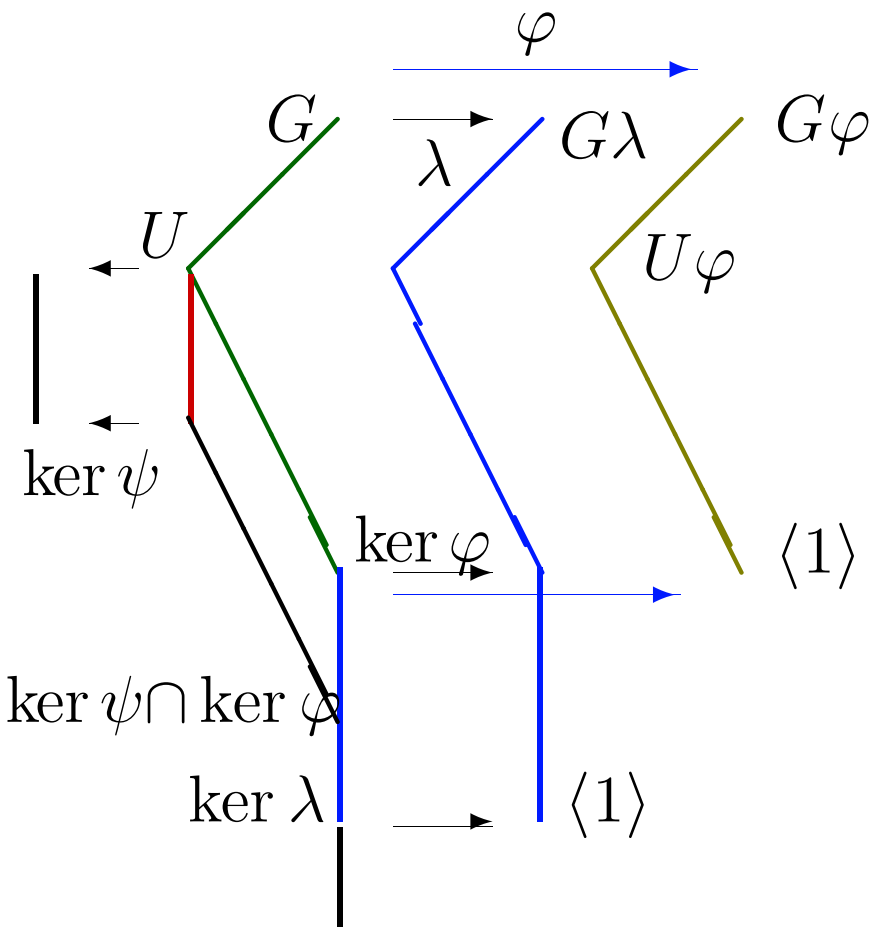


Given G and φ:

- Take a subgroup U of $G\varphi$.
- Rewrite the presentation (Consider U as another finitely presented group)
- Find quotients ψ of U.
- Induce ψ to λ defined on G.

λ tells more about the structure of G than ψ does.

Finding new Quotients



Given G and φ:

- Take a subgroup U of $G\varphi$.
- Rewrite the presentation (Consider U as another finitely presented group)
- Find quotients ψ of U.
- Induce ψ to λ defined on G.

λ tells more about the structure of G than ψ does.

The separate techniques are known, but the induction needs good permutation group functionality to be practical.

Application to Quotients

If $\ker \lambda \neq \ker \varphi$ we have found a bigger quotient.

Application to Quotients

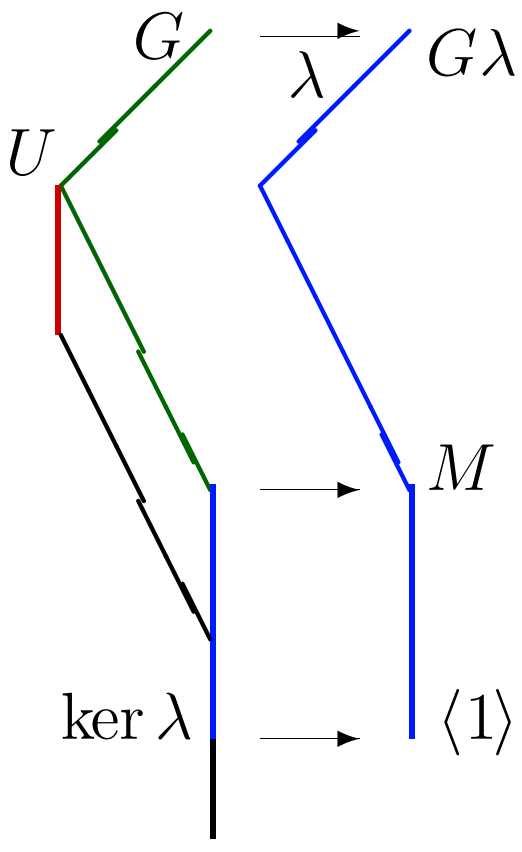
If $\ker \lambda \neq \ker \varphi$ we have found a bigger quotient.

Question: What subgroups U to take to find good quotients?

Application to Quotients

If $\ker \lambda \neq \ker \varphi$ we have found a bigger quotient.

Question: What subgroups U to take to find good quotients?

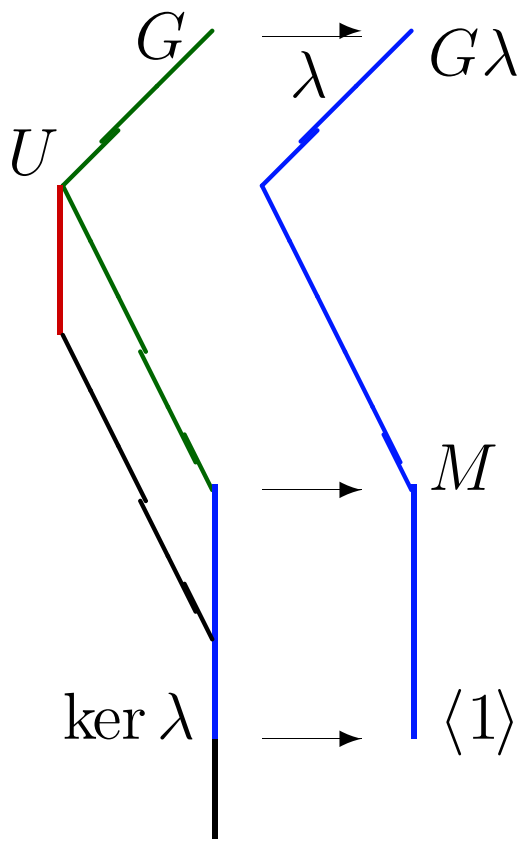


Assume: $U\psi$ is vector space (abelian quotient)

Application to Quotients

If $\ker \lambda \neq \ker \varphi$ we have found a bigger quotient.

Question: What subgroups U to take to find good quotients?



Assume: $U\psi$ is vector space (abelian quotient)

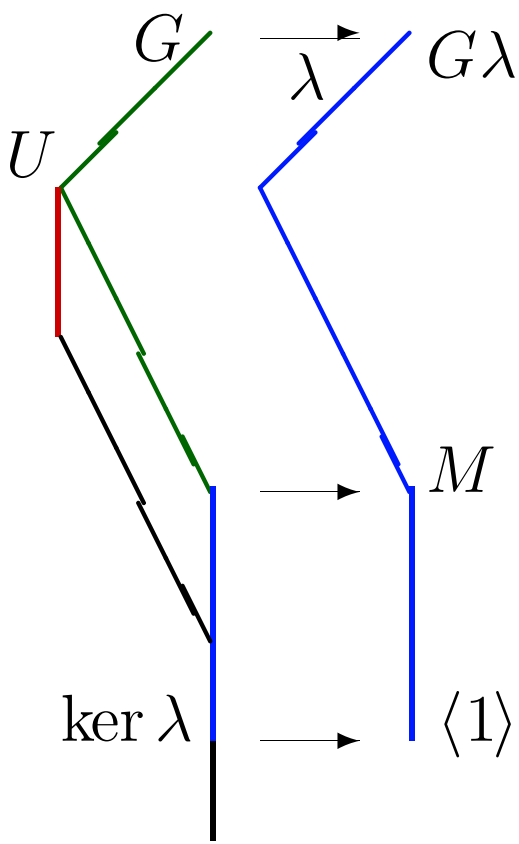
Then: $G\lambda$ will be an extension of a vector space M by $G\varphi$.

M is a *module* for G and $G\varphi$.

Application to Quotients

If $\ker\lambda \neq \ker\varphi$ we have found a bigger quotient.

Question: What subgroups U to take to find good quotients?



Assume: $U\psi$ is vector space (abelian quotient)

Then: $G\lambda$ will be an extension of a vector space M by $G\varphi$.

M is a *module* for G and $G\varphi$.

Criterion: To find a simple module M, $U\varphi$ must be a vector stabilizer for the *dual* module M^*.

Vice versa

If $U\varphi$ is a stabilizer for the dual module, either

- M will be exposed by an abelian quotient of U

Vice versa

If $U\varphi$ is a stabilizer for the dual module, either

- M will be exposed by an abelian quotient of U
- or $U\lambda$ has a quotient that is a *Schurian* extension by $U\varphi$.

Vice versa

If $U\varphi$ is a stabilizer for the dual module, either

- M will be exposed by an abelian quotient of U
- or $U\lambda$ has a quotient that is a *Schurian* extension by $U\varphi$.

The second case is related to the multiplier of $U\varphi$ and one can test in a Darstellungsgruppe whether it occurs.

For Example

BAUMEISTER/IVANOV/PASECHNIK (2000) give a presentation for a group G and $G\varphi \cong McL$. ($|McL| = 898\,128\,000$)

For Example

BAUMEISTER/IVANOV/PASECHNIK (2000) give a presentation for a group G and $G\varphi \cong McL$. ($|McL| = 898\,128\,000$)

Induction of subgroup quotients (in GAP) finds a lift $3^{104}.McL$ of degree $92\,400$ and a different lift $3^{127}.McL$ of degree $231\,000$.

The combination lift $3^{231}.McL$ is the largest lift for modules of dimension up to 600.

For Example

BAUMEISTER/IVANOV/PASECHNIK (2000) give a presentation for a group G and $G\varphi \cong McL$. ($|McL| = 898\,128\,000$)

Induction of subgroup quotients (in GAP) finds a lift $3^{104}.McL$ of degree $92\,400$ and a different lift $3^{127}.McL$ of degree $231\,000$.

The combination lift $3^{231}.McL$ is the largest lift for modules of dimension up to 600.

The permutation group calculations in the image group can become hard.

Towards a Nonsolvable Quotient Algorithm

(Joint work with A. NIEMEYER, Perth)

Given $G = \langle \underline{\mathbf{g}} \mid \mathcal{R} \rangle$ and $\varphi\colon G \to B$ for a finite B, find the largest lift of φ by a module.

Towards a Nonsolvable Quotient Algorithm

(Joint work with A. NIEMEYER, Perth)

Given $G = \langle \underline{\mathbf{g}} \mid \mathcal{R} \rangle$ and $\varphi\colon G \to B$ for a finite B, find the largest lift of φ by a module.

Idea

(In generalization of the Solvable Quotient Algorithm):

Find a presentation for B in the images of the generators $\underline{\mathbf{g}}$ under φ.

The relators of this presentation evaluate in G to elements of $\ker \varphi$.

These values become module generators for M.

Towards a Nonsolvable Quotient Algorithm

(Joint work with A. NIEMEYER, Perth)

Given $G = \langle \underline{\mathbf{g}} \mid \mathcal{R} \rangle$ and $\varphi\colon G \to B$ for a finite B, find the largest lift of φ by a module.

Idea

(In generalization of the Solvable Quotient Algorithm):

Find a presentation for B in the images of the generators $\underline{\mathbf{g}}$ under φ.

The relators of this presentation evaluate in G to elements of $\ker \varphi$.

These values become module generators for M.

Involve the group relators $\mathcal{R}$ to ensure it is a quotient of G.

Concretely

Construct a confluent rewriting system for $M.B$.

Start with a confluent rewriting system for B:

$$\langle x_1, \ldots, x_n \mid l_1(\underline{\mathbf{x}}) \to r_1(\underline{\mathbf{x}}), \ldots, l_m(\underline{\mathbf{x}}) \to r_m(\underline{\mathbf{x}}) \rangle.$$

Modify this rewriting system suitably. This can be considered as a “next order approximation" of G.

Concretely

Construct a confluent rewriting system for $M.B$.

Start with a confluent rewriting system for B:

$$\langle x_1, \ldots, x_n \mid l_1(\underline{\mathbf{x}}) \to r_1(\underline{\mathbf{x}}), \ldots, l_m(\underline{\mathbf{x}}) \to r_m(\underline{\mathbf{x}}) \rangle.$$

Modify this rewriting system suitably. This can be considered as a "next order approximation" of G.

The rewriting system for $M.B$ will involve generators $\underline{\mathbf{y}}$ that stand for coset representatives corresponding to $\underline{\mathbf{x}}$, and extra generators $\underline{\mathbf{z}}$ that represent elements of M.

Cofactors

In $M.B$ elements of B become coset representatives.

A product of coset representatives lies in a coset but is not necessarily its representative.

Cofactors

In $M.B$ elements of B become coset representatives.

A product of coset representatives lies in a coset but is not necessarily its representative. A rule

$$l_i(\underline{\mathbf{x}}) \longrightarrow r_i(\underline{\mathbf{x}})$$

therefore must be changed by a suitable element z_i of M:

$$l_i(\underline{\mathbf{y}}) \longrightarrow r_i(\underline{\mathbf{y}}) \cdot z_i.$$

Cofactors

In $M.B$ elements of B become coset representatives.

A product of coset representatives lies in a coset but is not necessarily its representative. A rule

$$l_i(\underline{\mathbf{x}}) \longrightarrow r_i(\underline{\mathbf{x}})$$

therefore must be changed by a suitable element z_i of M:

$$l_i(\underline{\mathbf{y}}) \longrightarrow r_i(\underline{\mathbf{y}}) \cdot z_i.$$

We initially consider these z_i to be variables whose values we have to find.

Action on M

As (representatives for B) act on M by conjugation we add extra generators $z_{i,w}$ for indices i and words w representing elements of B in normal form that stand for conjugates $z_i^{w(\underline{\mathbf{y}})}$.

Action on M

As (representatives for B) act on M by conjugation we add extra generators $z_{i,w}$ for indices i and words w representing elements of B in normal form that stand for conjugates $z_i^{w(\underline{\mathbf{y}})}$.

To work with these we formally add rules

$$z_{i,w} \cdot v \longrightarrow v \cdot z_{i,w \cdot v}$$

that specify this action as well as rules

$$z_{i,v} \cdot z_{j,w} \longrightarrow z_{j,w} \cdot z_{i,v} \quad \text{if } j < i \text{ or } v > w$$

and

$$z_{i,w}^p \longrightarrow 1$$

that indicate that M is elementary abelian.

Confluence

We never store these rules explicitly. Instead we use a modified algorithm for collection in the extension that takes care of these rules.

Confluence

We never store these rules explicitly. Instead we use a modified algorithm for collection in the extension that takes care of these rules.

To acchieve confluence, we look at all overlaps of left hand sides and collect them in both possible ways.

Confluence

We never store these rules explicitly. Instead we use a modified algorithm for collection in the extension that takes care of these rules.

To acchieve confluence, we look at all overlaps of left hand sides and collect them in both possible ways.

As the rules hold in B the result are relations that must hold among the $z_{i,w}$:

Assume $B = \langle x_1, x_2 \mid x_1^2 \to 1, x_2^2 \to 1, x_1x_2x_1 \to x_2x_1x_2 \rangle$.

New rules: ${y_1}^2 \to z_1, \quad , {y_2}^2 \to z_2, \quad y_1y_2y_1 \to y_2y_1y_2 \cdot z_3$.

$$
\begin{aligned}
\underline{y_1y_1}y_2y_1y_2y_1 \quad &\to \quad z_1 \cdot y_2y_1y_2y_1 \to y_2y_1y_2y_1 \cdot z_{1,y_2y_1y_2y_1} \\
&\to \quad y_2\underline{y_1y_2y_1}z_{1,y_1y_2} \to \underline{y_2y_2}y_1y_2 \cdot z_3 \cdot z_{1,y_1y_2} \\
&\to \quad z_2 \cdot y_1y_2 \cdot z_{1,y_1y_2} \cdot z_3 \to y_1y_2 \cdot z_{1,y_1y_2}z_{2,y_1y_2}z_3
\end{aligned}
$$

Assume $B = \langle x_1, x_2 \mid x_1^2 \to 1, x_2^2 \to 1, x_1x_2x_1 \to x_2x_1x_2 \rangle$.

New rules: ${y_1}^2 \to z_1, \quad , {y_2}^2 \to z_2, \quad y_1y_2y_1 \to y_2y_1y_2 \cdot z_3$.

$$
\begin{aligned}
\underline{y_1y_1}y_2y_1y_2y_1 &\to z_1 \cdot y_2y_1y_2y_1 \to y_2y_1y_2y_1 \cdot z_{1,y_2y_1y_2y_1} \\
&\to y_2\underline{y_1y_2y_1}z_{1,y_1y_2} \to \underline{y_2y_2}y_1y_2 \cdot z_3 \cdot z_{1,y_1y_2} \\
&\to z_2 \cdot y_1y_2 \cdot z_{1,y_1y_2} \cdot z_3 \to y_1y_2 \cdot z_{1,y_1y_2}z_{2,y_1y_2}z_3 \\
y_1\underline{y_1y_2y_1}y_2y_1 &\to y_1y_2y_1y_2 \cdot z_3 \cdot y_2y_1 \to y_1y_2y_1\underline{y_2y_2}y_1 \cdot z_{3,y_2y_1} \\
&\to y_1y_2y_1 \cdot z_2 \cdot y_1 \cdot z_{3,y_2y_1} \to y_1y_2\underline{y_1y_1} \cdot z_{2,y_1} \cdot z_{3,y_2y_1} \\
&\to y_1y_2 \cdot z_1 \cdot z_{2,y_1} \cdot z_{3,y_2y_1}
\end{aligned}
$$

Assume $B = \langle x_1, x_2 \mid x_1^2 \to 1, x_2^2 \to 1, x_1x_2x_1 \to x_2x_1x_2 \rangle$.

New rules: ${y_1}^2 \to z_1, \quad , {y_2}^2 \to z_2, \quad y_1y_2y_1 \to y_2y_1y_2 \cdot z_3$.

$$
\begin{aligned}
\underline{y_1y_1}y_2y_1y_2y_1 &\to z_1 \cdot y_2y_1y_2y_1 \to y_2y_1y_2y_1 \cdot z_{1,y_2y_1y_2y_1} \\
&\to y_2\underline{y_1y_2y_1}z_{1,y_1y_2} \to \underline{y_2y_2}y_1y_2 \cdot z_3 \cdot z_{1,y_1y_2} \\
&\to z_2 \cdot y_1y_2 \cdot z_{1,y_1y_2} \cdot z_3 \to y_1y_2 \cdot z_{1,y_1y_2}z_{2,y_1y_2}z_3 \\
y_1\underline{y_1y_2y_1}y_2y_1 &\to y_1y_2y_1y_2 \cdot z_3 \cdot y_2y_1 \to y_1y_2y_1\underline{y_2y_2}y_1 \cdot z_{3,y_2y_1} \\
&\to y_1y_2y_1 \cdot z_2 \cdot y_1 \cdot z_{3,y_2y_1} \to y_1y_2\underline{y_1y_1} \cdot z_{2,y_1} \cdot z_{3,y_2y_1} \\
&\to y_1y_2 \cdot z_1 \cdot z_{2,y_1} \cdot z_{3,y_2y_1}
\end{aligned}
$$

Deduction: $z_1 \cdot z_{2,y_1} \cdot z_{3,y_2y_1} = z_{1,y_1y_2} \cdot z_{2,y_1y_2} \cdot z_3$

Assume $B = \langle x_1, x_2 \mid x_1^2 \to 1, x_2^2 \to 1, x_1x_2x_1 \to x_2x_1x_2 \rangle$.

New rules: ${y_1}^2 \to z_1, \quad , {y_2}^2 \to z_2, \quad y_1y_2y_1 \to y_2y_1y_2 \cdot z_3$.

$$
\begin{aligned}
\underline{y_1y_1}y_2y_1y_2y_1 &\to z_1 \cdot y_2y_1y_2y_1 \to y_2y_1y_2y_1 \cdot z_{1,y_2y_1y_2y_1} \\
&\to y_2\underline{y_1y_2y_1}z_{1,y_1y_2} \to \underline{y_2y_2}y_1y_2 \cdot z_3 \cdot z_{1,y_1y_2} \\
&\to z_2 \cdot y_1y_2 \cdot z_{1,y_1y_2} \cdot z_3 \to y_1y_2 \cdot z_{1,y_1y_2}z_{2,y_1y_2}z_3 \\
y_1\underline{y_1y_2y_1}y_2y_1 &\to y_1y_2y_1y_2 \cdot z_3 \cdot y_2y_1 \to y_1y_2y_1\underline{y_2y_2}y_1 \cdot z_{3,y_2y_1} \\
&\to y_1y_2y_1 \cdot z_2 \cdot y_1 \cdot z_{3,y_2y_1} \to y_1y_2\underline{y_1y_1} \cdot z_{2,y_1} \cdot z_{3,y_2y_1} \\
&\to y_1y_2 \cdot z_1 \cdot z_{2,y_1} \cdot z_{3,y_2y_1}
\end{aligned}
$$

Deduction: $z_1 \cdot z_{2,y_1} \cdot z_{3,y_2y_1} = z_{1,y_1y_2} \cdot z_{2,y_1y_2} \cdot z_3$

Once all deduced relations hold, we have rules for and extension of a sum of regular modules.

Images

The epimorphism $\varphi\colon G \to B$ is specified by the images of the generators g_i of G.

We write those in normal form for the rewriting system for B:

$$\varphi(g_i) = d_i(\underline{\mathbf{x}})$$

Images

The epimorphism $\varphi\colon G \to B$ is specified by the images of the generators g_i of G.

We write those in normal form for the rewriting system for B:

$$\varphi(g_i) = d_i(\underline{\mathbf{x}})$$

Under λ the images agree on the coset representatives of M but the group elements may differ by elements of M:

$$\lambda(g_i) = d_i(\underline{\mathbf{y}}) \cdot z_{i+m}$$

(m=number of rules for B)

Again, we add extra variable generators $z_{i+m,w}$ to represent these elements.

Enforcing Quotient Property

To enforce the relations $\mathcal{R}$ of G, we evaluate them in the images $\lambda(g_i)$ and compute normal forms in the extended rewriting system.

Enforcing Quotient Property

To enforce the relations $\mathcal{R}$ of G, we evaluate them in the images $\lambda(g_i)$ and compute normal forms in the extended rewriting system.

For example if a relator is $(x_1 x_2)^2$, we evaluate $\left(d_1(\underline{\mathbf{y}}) \cdot z_{1+m} \cdot d_2(\underline{\mathbf{y}}) \cdot z_{2+m}\right)^2$ and collect.

Enforcing Quotient Property

To enforce the relations $\mathcal{R}$ of G, we evaluate them in the images $\lambda(g_i)$ and compute normal forms in the extended rewriting system.

For example if a relator is $(x_1 x_2)^2$, we evaluate $\left(d_1(\underline{\mathbf{y}}) \cdot z_{1+m} \cdot d_2(\underline{\mathbf{y}}) \cdot z_{2+m}\right)^2$ and collect.

As B is an image, these relators must hold in the elements $\varphi(g_i)$, thus the result of the collection is a relation among the $z_{i,w}$.

We remember these relations.

Choice of Representatives

λ can change by choosing different coset representatives in $M.B$ for the elements of B.

To remove this choice, we add "definition rules":

Choice of Representatives

λ can change by choosing different coset representatives in $M.B$ for the elements of B.

To remove this choice, we add "definition rules":

In B we can express the generators corresponding to the x_i as words in the $\varphi(g_i)$.

We use the *same words*, evaluated in the $\lambda(g_i)$, as *definition* of the generators of $G\lambda$ (corresponding to the symbols y_i).

Choice of Representatives

λ can change by choosing different coset representatives in $M.B$ for the elements of B.

To remove this choice, we add "definition rules":

In B we can express the generators corresponding to the x_i as words in the $\varphi(g_i)$.

We use the *same words*, evaluated in the $\lambda(g_i)$, as *definition* of the generators of $G\lambda$ (corresponding to the symbols y_i).

(For experts: This is the main difference to the Solvable Quotient algorithm)

Collecting those definitions gives further relations among the $z_{i,w}$.

Module Presentation

We end up with (many) relations among B-conjugates of the z_i. We can consider this as a B-*Module presentation* for M.

It is a quotient of the free module generated by the $z_i = z_{i,1}$.

Module Presentation

We end up with (many) relations among B-conjugates of the z_i. We can consider this as a B-*Module presentation* for M.

It is a quotient of the free module generated by the $z_i = z_{i,1}$.

There is an algorithm called *module enumeration* (S. LINTON) that will take such a module presentation and compute:

- A module basis.

Module Presentation

We end up with (many) relations among B-conjugates of the z_i. We can consider this as a B-*Module presentation* for M.

It is a quotient of the free module generated by the $z_i = z_{i,1}$.

There is an algorithm called *module enumeration* (S. LINTON) that will take such a module presentation and compute:

- A module basis.
- Coefficients of the z_i.

Module Presentation

We end up with (many) relations among B-conjugates of the z_i. We can consider this as a B-*Module presentation* for M.

It is a quotient of the free module generated by the $z_i = z_{i,1}$.

There is an algorithm called *module enumeration* (S. LINTON) that will take such a module presentation and compute:

- A module basis.
- Coefficients of the z_i.
- Matrices for the action on this basis.

Module Presentation

We end up with (many) relations among B-conjugates of the z_i. We can consider this as a B-*Module presentation* for M.

It is a quotient of the free module generated by the $z_i = z_{i,1}$.

There is an algorithm called *module enumeration* (S. LINTON) that will take such a module presentation and compute:

- A module basis.
- Coefficients of the z_i.
- Matrices for the action on this basis.

(Can one use noncommutative Gröbner bases?)

We take this result and feed it back in the extended rewriting system to replace the $z_{i,w}$. If $\underline{\mathbf{b}}$ are symbols for the basis, this yields a confluent rewriting system (with respect to a wreath product ordering) for $M.B$:

$$
\begin{aligned}
l_i(\underline{\mathbf{y}}) &\rightarrow r_i(\underline{\mathbf{y}}) \cdot w_i(\underline{\mathbf{b}}) \\
b_i y_j &\rightarrow y_j \cdot y_j w(\underline{\mathbf{b}}) \qquad \text{(matrix action)} \\
b_i b_j &\rightarrow b_j b_i \qquad (j < i), \qquad\qquad b_i^p \rightarrow 1
\end{aligned}
$$

We take this result and feed it back in the extended rewriting system to replace the $z_{i,w}$. If $\underline{\mathbf{b}}$ are symbols for the basis, this yields a confluent rewriting system (with respect to a wreath product ordering) for $M.B$:

$$\begin{aligned} l_i(\underline{\mathbf{y}}) &\rightarrow r_i(\underline{\mathbf{y}}) \cdot w_i(\underline{\mathbf{b}}) \\ b_i y_j &\rightarrow y_j \cdot y_j w(\underline{\mathbf{b}}) \qquad \text{(matrix action)} \\ b_i b_j &\rightarrow b_j b_i \qquad (j < i), \qquad\qquad b_i^p \rightarrow 1 \end{aligned}$$

We also can deduce the epimorphism λ and know that the group is a quotient of G.

And then...

The output information (rewriting system/epimorphism/decomposition of generators in images) is sufficient to iterate in search of another lift.

And then...

The output information (rewriting system/epimorphism/decomposition of generators in images) is sufficient to iterate in search of another lift.

I have a trial implementation of a single step to play with.

Example: The Heineken Group

$$G = \langle x, y, z \mid [x,[x,y]] = z, [y,[y,z]] = x, [z,[z,x]] = y \rangle$$

Example: The Heineken Group

$$G = \langle x, y, z \mid [x,[x,y]] = z, [y,[y,z]] = x, [z,[z,x]] = y \rangle$$

- G is perfect: No solvable quotients

Example: The Heineken Group

$$G = \langle x, y, z \mid [x,[x,y]] = z, [y,[y,z]] = x, [z,[z,x]] = y \rangle$$

- G is perfect: No solvable quotients
- $\varphi\colon x \mapsto (1,2,5,4,3)$, $y \mapsto (1,2,3,5,4)$, $z \mapsto (1,2,4,3,5)$ is an epimorphism onto A_5.

If one run the algorithm for $p = 2$ we find that $M = 2^5$. This exposes a quotient $2^5.A_5$.

Example: The Heineken Group

$$G = \langle x, y, z \mid [x,[x,y]] = z, [y,[y,z]] = x, [z,[z,x]] = y \rangle$$

- G is perfect: No solvable quotients
- $\varphi\colon x \mapsto (1,2,5,4,3)$, $y \mapsto (1,2,3,5,4)$, $z \mapsto (1,2,4,3,5)$ is an epimorphism onto A_5.

If one run the algorithm for $p = 2$ we find that $M = 2^5$. This exposes a quotient $2^5.A_5$.

The second step failed on my computer – the vector enumeration got too big. (The group is known to be infinite)

Example: The Heineken Group

$$G = \langle x, y, z \mid [x,[x,y]] = z, [y,[y,z]] = x, [z,[z,x]] = y \rangle$$

- G is perfect: No solvable quotients
- $\varphi\colon x \mapsto (1,2,5,4,3)$, $y \mapsto (1,2,3,5,4)$, $z \mapsto (1,2,4,3,5)$ is an epimorphism onto A_5.

If one run the algorithm for $p = 2$ we find that $M = 2^5$. This exposes a quotient $2^5.A_5$.

The second step failed on my computer – the vector enumeration got too big. (The group is known to be infinite)

Example: The Heineken Group

$$G = \langle x, y, z \mid [x,[x,y]] = z, [y,[y,z]] = x, [z,[z,x]] = y \rangle$$

- G is perfect: No solvable quotients
- $\varphi\colon x \mapsto (1,2,5,4,3)$, $y \mapsto (1,2,3,5,4)$, $z \mapsto (1,2,4,3,5)$ is an epimorphism onto A_5.

If one run the algorithm for $p = 2$ we find that $M = 2^5$. This exposes a quotient $2^5.A_5$.

The second step failed on my computer – the vector enumeration got too big. (The group is known to be infinite)

This algorithm is still very experimental and will need a lot of fine-tuning.

Summary

- Representation of Subgroups.

Summary

- Representation of Subgroups.
- Methods (Attempts) to lift homomorphisms.

Summary

- Representation of Subgroups.
- Methods (Attempts) to lift homomorphisms.
- Advantages of having a system