

# Groebner Bases and related methods in Group Theory

Alexander Hulpke

Department of Mathematics  
Colorado State University  
Fort Collins, CO, 80523-1874

Workshop D1, Linz, 5/4/06

**THE FOLLOWING **PREVIEW** HAS BEEN APPROVED FOR  
ALL AUDIENCES**

**BY THE MOTION PICTURE ASSOCIATION OF AMERICA**

**THE FILM ADVERTISED HAS BEEN RATED**

<b>PG-13</b>	<b>PARENTS STRONGLY CAUTIONED</b> 
<b>Some Material May Be Inappropriate for Children Under 13</b>	
<b>INTENSE THEMATIC MATERIAL, SEXUAL CONTENT AND A SCENE OF VIOLENCE</b>	

GAP <http://www.gap-system.org> is a free and open system easily described as “Maple for Discrete Mathematics”.

It started out for group theory, but now contains much of related areas.

- Comfortable programming environment for mathematics
- Reasonably efficient implementations of many basic algorithms.
- Many packages provide extensions.

Ask me for more details.

Some examples on the way.

GAP <http://www.gap-system.org> is a free and open system easily described as “Maple for Discrete Mathematics”.

It started out for group theory, but now contains much of related areas.

- Comfortable programming environment for mathematics
- Reasonably efficient implementations of many basic algorithms.
- Many packages provide extensions.

Ask me for more details.

Some examples on the way.

GAP <http://www.gap-system.org> is a free and open system easily described as “Maple for Discrete Mathematics”.

It started out for group theory, but now contains much of related areas.

- Comfortable programming environment for mathematics
- Reasonably efficient implementations of many basic algorithms.
- Many packages provide extensions.

Ask me for more details.

Some examples on the way.

# Two ways of specifying a structure

- By elements or generators (basis, points in a variety).
- By relations (nullspace, ideal of a variety).

Relations are easy to write down, but it can be hard to find all solutions.

# Finitely Presented Groups

The analogous idea for groups is that of a finitely presented group:

**Alphabet** A finite set of symbols.

**Free Group** Words (including the empty word) in an alphabet, including formal inverse letters, cancellation of  $xx^{-1}$  and  $x^{-1}x$ .

**Finitely presented group** Factor of a free group by relations (or relators:  $a = b$  implies  $ab^{-1} = 1$ ).

# Example

You probably have seen such groups in a first abstract algebra course.

$$D_8 = \langle r, s \mid r^2 = s^4 = 1, s^r = s^{-1} \rangle.$$

In general we have finitely many relators.

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

(A **bold** letter  $\underline{x}$  denotes a set  $\{x_1, \dots, x_n\}$ ).

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

# Example

You probably have seen such groups in a first abstract algebra course.

$$D_8 = \langle r, s \mid r^2 = s^4 = 1, s^r = s^{-1} \rangle.$$

In general we have finitely many relators.

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

(A **bold** letter  $\underline{\mathbf{x}}$  denotes a set  $\{x_1, \dots, x_n\}$ ).

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

# Example

You probably have seen such groups in a first abstract algebra course.

$$D_8 = \langle r, s \mid r^2 = s^4 = 1, s^r = s^{-1} \rangle.$$

In general we have finitely many relators.

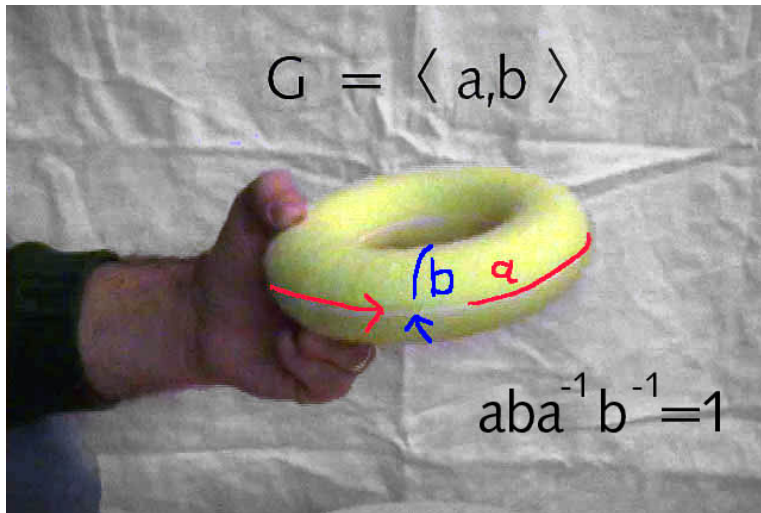
$$\langle \underline{\mathbf{x}} \mid \mathcal{R} \rangle = \langle x_1, \dots, x_n \mid l_1(\underline{\mathbf{x}}) = r_1(\underline{\mathbf{x}}), \dots, l_m(\underline{\mathbf{x}}) = r_m(\underline{\mathbf{x}}) \rangle.$$

(A **bold** letter  $\underline{\mathbf{x}}$  denotes a set  $\{x_1, \dots, x_n\}$ ).

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

# Where do they come from?

Presentations arise in topology as the natural way to represent a fundamental group.



Consider the symmetries of a square:

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

Every element can be brought in the form  $r^e s^f$  with  $0 \leq e \leq 1$ ,  $0 \leq f \leq 3$ .

(This is called collection.)

We thus have 8 elements in total.

Consider the symmetries of a square:

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

Every element can be brought in the form  $r^e s^f$  with  $0 \leq e \leq 1$ ,  $0 \leq f \leq 3$ .

(This is called collection.)

We thus have 8 elements in total.

Consider the symmetries of a square:

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

Every element can be brought in the form  $r^e s^f$  with  $0 \leq e \leq 1$ ,  
 $0 \leq f \leq 3$ .

(This is called collection.)

We thus have 8 elements in total.

# Alas

(What the textbook doesn't tell you)

This does not always work:

- A rule of the form  $sr = rs + \text{decoration}$  implies that there is a normal subgroup  $N \triangleleft G$  such that  $G/N = \langle Nr \rangle$  is cyclic.
- The order calculation we did does not always work:  
 $\langle r, s \mid r^5 = s^2 = 1, rs = sr^3 \rangle$  has order 2, not 5.
- The rules given might not easily give a normal form:  
 $\langle a, b \mid ab^2a^2 = a(b^2a^3)^4 = 1 \rangle$ .
- What happens if we can apply several rules in one situation?

# Alas

(What the textbook doesn't tell you)

This does not always work:

- A rule of the form  $sr = rs + \text{decoration}$  implies that there is a normal subgroup  $N \triangleleft G$  such that  $G/N = \langle Nr \rangle$  is cyclic.
- The order calculation we did does not always work:  
 $\langle r, s \mid r^5 = s^2 = 1, rs = sr^3 \rangle$  has order 2, not 5.
- The rules given might not easily give a normal form:  
 $\langle a, b \mid ab^2a^2 = a(b^2a^3)^4 = 1 \rangle$ .
- What happens if we can apply several rules in one situation?

# Alas

(What the textbook doesn't tell you)

This does not always work:

- A rule of the form  $sr = rs + \text{decoration}$  implies that there is a normal subgroup  $N \triangleleft G$  such that  $G/N = \langle Nr \rangle$  is cyclic.
- The order calculation we did does not always work:  
 $\langle r, s \mid r^5 = s^2 = 1, rs = sr^3 \rangle$  has order 2, not 5.
- The rules given might not easily give a normal form:  
 $\langle a, b \mid ab^2a^2 = a(b^2a^3)^4 = 1 \rangle$ .
- What happens if we can apply several rules in one situation?

# Alas

(What the textbook doesn't tell you)

This does not always work:

- A rule of the form  $sr = rs + \text{decoration}$  implies that there is a normal subgroup  $N \triangleleft G$  such that  $G/N = \langle Nr \rangle$  is cyclic.
- The order calculation we did does not always work:  
 $\langle r, s \mid r^5 = s^2 = 1, rs = sr^3 \rangle$  has order 2, not 5.
- The rules given might not easily give a normal form:  
 $\langle a, b \mid ab^2a^2 = a(b^2a^3)^4 = 1 \rangle$ .
- What happens if we can apply several rules in one situation?

# Even worse

It has been shown that it is in general **algorithmically impossible** to decide in bounded time that a finitely presented group is trivial. (BOONE, NOVIKOV, 1956: Translate to Halteproblem for Turing machine.)

This means we are not even guaranteed to be able to check whether two words represent the same element.

All methods for finitely presented groups therefore either are “opportunistic” or only expose certain quotient groups which are not guaranteed to be faithful.

In many of these methods Gröbner basis-like methods play a crucial rôle.

# Even worse

It has been shown that it is in general **algorithmically impossible** to decide in bounded time that a finitely presented group is trivial. (BOONE, NOVIKOV, 1956: Translate to Halteproblem for Turing machine.)

This means we are not even guaranteed to be able to check whether two words represent the same element.

All methods for finitely presented groups therefore either are “opportunistic” or only expose certain quotient groups which are not guaranteed to be faithful.

In many of these methods Gröbner basis-like methods play a crucial rôle.

# Even worse

It has been shown that it is in general **algorithmically impossible** to decide in bounded time that a finitely presented group is trivial. (BOONE, NOVIKOV, 1956: Translate to Halteproblem for Turing machine.)

This means we are not even guaranteed to be able to check whether two words represent the same element.

All methods for finitely presented groups therefore either are “opportunistic” or only expose certain quotient groups which are not guaranteed to be faithful.

In many of these methods Gröbner basis-like methods play a crucial rôle.

# Homomorphisms

Testing whether a map  $\varphi: G \rightarrow H$  is a homomorphism is easy, once we can compare elements in  $H$ :

*For every relator  $r(\underline{\mathbf{g}})$  of  $G$ , we must have  $r(\underline{\mathbf{g}}^\varphi) = 1_H$ .*

A principal tool for studying finitely presented groups therefore is to find quotient groups.

This is what we want to do.

# Homomorphisms

Testing whether a map  $\varphi: G \rightarrow H$  is a homomorphism is easy, once we can compare elements in  $H$ :

*For every relator  $r(\underline{\mathbf{g}})$  of  $G$ , we must have  $r(\underline{\mathbf{g}}^\varphi) = 1_H$ .*

A principal tool for studying finitely presented groups therefore is to find quotient groups.

This is what we want to do.

# Abelian Quotients

The easiest case is that of abelian quotients.

We can find these by abelianizing the presentation, writing it in matrix form, and transforming it into (Smith) normal form:

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

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

# Abelian Quotients

The easiest case is that of abelian quotients.

We can find these by abelianizing the presentation, writing it in matrix form, and transforming it into (Smith) normal form:

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

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

# Abelian Quotients

The easiest case is that of abelian quotients.

We can find these by abelianizing the presentation, writing it in matrix form, and transforming it into (Smith) normal form:

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

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

# Abelian Quotients

The easiest case is that of abelian quotients.

We can find these by abelianizing the presentation, writing it in matrix form, and transforming it into (Smith) normal form:

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

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

# Matrix Group Quotients

(PLESKEN, SOUVIGNIER, ROBERTZ)

Suppose  $G = \langle \underline{\mathbf{g}} \mid R \rangle$  is a finitely presented group and we want to find a homomorphism  $\varphi: G \rightarrow GL_n(F)$ .

For this we consider the images of the generators as matrices with variable entries:

$$\mathbf{g}_i^\varphi = \begin{pmatrix} a_{i,1,1} & a_{i,1,2} & \cdots \\ a_{i,2,1} & a_{i,2,2} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

The Relators  $r(\underline{\mathbf{g}})$  evaluated in these images now yield *polynomial* equations in the  $a_{i,j,k}$ .

The image of every  $n$ -dimensional representation then is in the corresponding variety.

# Matrix Group Quotients

(PLESKEN, SOUVIGNIER, ROBERTZ)

Suppose  $G = \langle \underline{\mathbf{g}} \mid R \rangle$  is a finitely presented group and we want to find a homomorphism  $\varphi: G \rightarrow GL_n(F)$ .

For this we consider the images of the generators as matrices with variable entries:

$$g_i^\varphi = \begin{pmatrix} a_{i,1,1} & a_{i,1,2} & \cdots \\ a_{i,2,1} & a_{i,2,2} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

The Relators  $r(\underline{\mathbf{g}})$  evaluated in these images now yield *polynomial* equations in the  $a_{i,j,k}$ .

The image of every  $n$ -dimensional representation then is in the corresponding variety.

# Removing Redundancies

There is however much duplication (for  $x \in GL_n(F)$  also  $\varphi^x$  is in the variety) as well as noninvertible matrices.

To reduce this problem we make a few conventions about the natural module  $M := F^n$ :

- $M$  is a simple module. (Otherwise we find all composition factors and then consider module extensions.)
- We assume the basis to be chosen as images of one vector under “short” products of group elements.

(PLESKEN et. al. use a method they attribute to JANET (1929) and GERDT (2001) for solving the polynomial equations.)

# Removing Redundancies

There is however much duplication (for  $x \in GL_n(F)$  also  $\varphi^x$  is in the variety) as well as noninvertible matrices.

To reduce this problem we make a few conventions about the natural module  $M := F^n$ :

- $M$  is a simple module. (Otherwise we find all composition factors and then consider module extensions.)
- We assume the basis to be chosen as images of one vector under “short” products of group elements.

(PLESKEN et. al. use a method they attribute to JANET (1929) and GERDT (2001) for solving the polynomial equations.)

# For example

Suppose that  $n = 2$ , we have that  $M = \langle v_1, v_2 \rangle$  and that  $G = \langle a, b \rangle$ . Then either  $v_1^a \notin \langle v_1 \rangle$  or  $v_1^b \notin \langle v_1 \rangle$ . (Otherwise  $\langle v_1 \rangle$  is a submodule.) Thus we get the following possible matrix images:

$$a \mapsto \begin{pmatrix} 0 & 1 \\ a_{2,1} & a_{2,2} \end{pmatrix} \quad b \mapsto \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix}$$

or

$$a \mapsto \begin{pmatrix} a_{1,1} & 0 \\ a_{2,1} & a_{2,2} \end{pmatrix} \quad b \mapsto \begin{pmatrix} 0 & 1 \\ b_{2,1} & b_{2,2} \end{pmatrix}$$

# Example Calculation

Let  $G = \langle a, b \mid a^2 = b^3 = (ab)^7 = 1 \rangle$  (Hurwitz group). We want to find representations in  $GL_2(8)$ .

```
gap> f:=FreeGroup("a","b");
<free group on the generators [ a, b ]>
gap> AssignGeneratorVariables(f);
#I Assigned the global variables [ a, b ]
gap> rels:=[a^2,b^3,(a*b)^7];
[ a^2, b^3, a*b*a*b*a*b*a*b*a*b*a*b*a*b ]
gap> g:=f/rels;
<fp group on the generators [ a, b ]>
gap> a11:=X(GF(8),"a11");;a21:=X(GF(8),"a21");;a22:=
gap> b11:=X(GF(8),"b11");;b12:=X(GF(8),"b12");;
gap> b21:=X(GF(8),"b21");;b22:=X(GF(8),"b22");;
```

```

gap> ai:=[[0,1],[a21,a22]]*One(GF(8));
[ [ 0*Z(2), Z(2)^0 ], [ a21, a22 ] ]
gap> bi:=[[b11,b12],[b21,b22]]*One(GF(8));
[ [ b11, b12 ], [ b21, b22 ] ]
gap> mr:=List(rels,
    i->MappedWord(i,[a,b],[ai,bi])-ai^0);
[ [ [ a21+Z(2)^0, a22 ], [ a21*a22, a22^2+a21+Z(2) ]
    ...
gap> polys:=List(Flat(mr),
    i->Value(i,[a21,a22],[Z(2),0*Z(2)]));
[ 0, 0, 0, 0, b11^3+b12*b21*b22+1,
    b11^2*b12+b11*b12*b22+b12^2*b21+b12*b22^2,
    ...

```

```

gap> rgb:=ReducedGroebnerBasis(polys,
      MonomialGrlexOrdering());
gap> rgb2:=ReducedGroebnerBasis(rgb,
      MonomialLexOrdering());
[ b21^6+b21^4*b22^2+b21^2*b22^4+b22^6+...
  ... b11+b22+1]
gap> List(rgb2,p->List(OccuringVariableIndices(p),i
[[b21,b22], [b12,b21,b22], [b12,b21,b22],
  [b12,b21,b22], [b12,b21,b22], [b11,b22]]
gap> eqs:=List(rgb2,
      i->Value(i,[b11,b22],[0*Z(2),Z(2)]));
[ b21^6+b21^5+b21^4+b21^3+b21^2+b21+Z(2)^0,
  ...
gap> Factors(PolynomialRing(GF(8)),eqs[1]);
[ b21+Z(2^3), b21+Z(2^3)^2, ... ]

```

```

gap> Value(eqs[2],[b21],[-Z(8)]);
b12+Z(2^3)^6
gap> mat1:=[[0,1],[1,0]]*One(GF(8));
gap> mat2:=[[0,-Z(8)^6],[-Z(8),1]]*One(GF(8));
gap> h:=Group(mat1,mat2);;Size(h);
504
gap> hom:=GroupHomomorphismByImages(g,h,
    GeneratorsOfGroup(g),[mat1,mat2]);
[a,b] -> [[0*Z(2),Z(2)^0],[Z(2)^0,0*Z(2)]],
    [[0*Z(2),Z(2^3)^6],[Z(2^3),Z(2)^0]]
gap> hom:=GroupHomomorphismByImages(g,h,
    GeneratorsOfGroup(g),[mat2,mat1]);
fail

```

# What about normal forms?

Let's go back to the "algebra textbook" example.

Can we always use relations to bring elements into "normal form"?

- We can transform equivalent words into each other, but any "length" might have to go up
- What means "normal form"? Apply rules as long as you can. We need some concept of ordering and "smallest".

# What about normal forms?

Let's go back to the "algebra textbook" example.

Can we always use relations to bring elements into "normal form"?

- We can transform equivalent words into each other, but any "length" might have to go up
- What means "normal form"? Apply rules as long as you can. We need some concept of ordering and "smallest".

# What about normal forms?

Let's go back to the "algebra textbook" example.

Can we always use relations to bring elements into "normal form"?

- We can transform equivalent words into each other, but any "length" might have to go up
- What means "normal form"? Apply rules as long as you can. We need some concept of ordering and "smallest".

# Rewriting Systems

## Analogy to Polynomial Division Algorithm

**Free Monoid:**  $F$  is the set of words in an alphabet, including empty word.

We consider a finitely presented monoid  $F/R$  where  $R$  is a set of relations of the form  $l = r$ . (As we don't have inverses we can't force  $lr^{-1}$ .)

As we really have a group, consider inverses as extra symbols, cancellation  $xx^{-1}$  becomes extra relations.

Also take total ordering  $<$  on  $F$  that is

Well ordering (no infinite descending chains – we want reductions to terminate)

Translation invariant  $a < b$  implies  $cad < cbd$ .

**Example:** Length+Lex. (Not pure lex!)

# Rewriting Systems

## Analogy to Polynomial Division Algorithm

**Free Monoid:**  $F$  is the set of words in an alphabet, including empty word.

We consider a finitely presented monoid  $F/R$  where  $R$  is a set of relations of the form  $l = r$ . (As we don't have inverses we can't force  $lr^{-1}$ .)

As we really have a group, consider inverses as extra symbols, cancellation  $xx^{-1}$  becomes extra relations.

Also take total ordering  $<$  on  $F$  that is

Well ordering (no infinite descending chains – we want reductions to terminate)

Translation invariant  $a < b$  implies  $cad < cbd$ .

**Example:** Length+Lex. (Not pure lex!)

# Rewriting Systems

## Analogy to Polynomial Division Algorithm

**Free Monoid:**  $F$  is the set of words in an alphabet, including empty word.

We consider a finitely presented monoid  $F/R$  where  $R$  is a set of relations of the form  $l = r$ . (As we don't have inverses we can't force  $lr^{-1}$ .)

As we really have a group, consider inverses as extra symbols, cancellation  $xx^{-1}$  becomes extra relations.

Also take total ordering  $\prec$  on  $F$  that is

**Well ordering** (no infinite descending chains – we want reductions to terminate)

**Translation invariant**  $a \prec b$  implies  $cad \prec cbd$ .

Example: Length+Lex. (Not pure lex!)

# Rewriting Systems

## Analogy to Polynomial Division Algorithm

**Free Monoid:**  $F$  is the set of words in an alphabet, including empty word.

We consider a finitely presented monoid  $F/R$  where  $R$  is a set of relations of the form  $l = r$ . (As we don't have inverses we can't force  $lr^{-1}$ .)

As we really have a group, consider inverses as extra symbols, cancellation  $xx^{-1}$  becomes extra relations.

Also take total ordering  $\prec$  on  $F$  that is

**Well ordering** (no infinite descending chains – we want reductions to terminate)

**Translation invariant**  $a \prec b$  implies  $cad \prec cbd$ .

**Example:** Length+Lex. (Not pure lex!)

# Rewriting Systems

## Analogy to Polynomial Division Algorithm

Now consider every relation  $l = r$  as directed  $l \rightarrow r$  (or  $r \rightarrow l$ ), reducing with respect to the ordering  $\prec$ .

Because we have a translation invariant well-ordering, we can apply rules only finitely often to a given word, yielding a not further reducible form.

## Potential Problems

Is this reduced form unique (Confluence)?

If it is, can we use it as a normal form (Church-Rosser property)?

Now consider every relation  $l = r$  as directed  $l \rightarrow r$  (or  $r \rightarrow l$ ),  
reducing with respect to the ordering  $\prec$ .

Because we have a translation invariant well-ordering, we can apply  
rules only finitely often to a given word, yielding a not further  
reducible form.

## Potential Problems

Is this reduced form unique (Confluence)?

If it is, can we use it as a normal form (Church-Rosser property)?

# The work of Knuth and Bendix

Analogy to Gröbner bases

## Theorem

*A rewriting system is confluent, if and only if it has the Church-Rosser property, if and only if all overlaps of left hand sides reduce in both ways to the same result.*

E.g.  $\langle x, y \mid x^2 = 1, y^2 = 1, xyx = yxy \rangle$ :

$$\underline{xx}yx = yx$$

$$\underline{xy}x = \underline{xy}y = \underline{yxy}y = yx$$

Analogous to Buchberger's algorithm this gives a method (Knuth-Bendix algorithm), to make a rewriting system confluent: If  $a \cdot b \cdot c$  reduces to  $p \cdot c$  and  $a \cdot q$  we add a rule  $p \cdot c \rightarrow a \cdot q$  to obtain equal reductions.

Then continue on other overlaps.

# The work of Knuth and Bendix

Analogy to Gröbner bases

## Theorem

*A rewriting system is confluent, if and only if it has the Church-Rosser property, if and only if all overlaps of left hand sides reduce in both ways to the same result.*

E.g.  $\langle x, y \mid x^2 = 1, y^2 = 1, xyx = yxy \rangle$ :

$$\underline{xx}yx = yx$$

$$\underline{xy}x = \underline{xy}y = \underline{yxy}y = yx$$

Analogous to Buchberger's algorithm this gives a method (Knuth-Bendix algorithm), to make a rewriting system confluent: If  $a \cdot b \cdot c$  reduces to  $p \cdot c$  and  $a \cdot q$  we add a rule  $p \cdot c \rightarrow a \cdot q$  to obtain equal reductions.

Then continue on other overlaps.

# The work of Knuth and Bendix

Analogy to Gröbner bases

## Theorem

*A rewriting system is confluent, if and only if it has the Church-Rosser property, if and only if all overlaps of left hand sides reduce in both ways to the same result.*

E.g.  $\langle x, y \mid x^2 = 1, y^2 = 1, xyx = yxy \rangle$ :

$$\underline{xx}yx = yx$$

$$\underline{xy}x = \underline{xy}y = yx\underline{yy} = yx$$

Analogous to Buchberger's algorithm this gives a method (Knuth-Bendix algorithm), to make a rewriting system confluent: If  $a \cdot b \cdot c$  reduces to  $p \cdot c$  and  $a \cdot q$  we add a rule  $p \cdot c \rightarrow a \cdot q$  to obtain equal reductions. Then continue on other overlaps.

# The work of Knuth and Bendix

Analogy to Gröbner bases

## Theorem

*A rewriting system is confluent, if and only if it has the Church-Rosser property, if and only if all overlaps of left hand sides reduce in both ways to the same result.*

E.g.  $\langle x, y \mid x^2 = 1, y^2 = 1, xyx = yxy \rangle$ :

$$\underline{xx}yx = yx$$

$$\underline{xy}x = \underline{xy}y = yx\underline{yy} = yx$$

Analogous to Buchberger's algorithm this gives a method (Knuth-Bendix algorithm), to make a rewriting system confluent: If  $a \cdot b \cdot c$  reduces to  $p \cdot c$  and  $a \cdot q$  we add a rule  $p \cdot c \rightarrow a \cdot q$  to obtain equal reductions.

Then continue on other overlaps.

# Difficulties

Not analogue to Gröbner bases

We cannot undo the Boone/Novikov result.

This means there cannot be any bound on the runtime (and memory used).

One can show that if the presentation defines a finite group, the process will terminate after finitely (but unbounded) many steps.

In practice one is often lucky.

# Difficulties

Not analogue to Gröbner bases

We cannot undo the Boone/Novikov result.

This means there cannot be any bound on the runtime (and memory used).

One can show that if the presentation defines a finite group, the process will terminate after finitely (but unbounded) many steps.

In practice one is often lucky.

# Difficulties

Not analogue to Gröbner bases

We cannot undo the Boone/Novikov result.

This means there cannot be any bound on the runtime (and memory used).

One can show that if the presentation defines a finite group, the process will terminate after finitely (but unbounded) many steps.

In practice one is often lucky.

# Example

$$\langle a, b \mid a^3 = b^3 = (ab)^3 = 1 \rangle$$

```
gap> f:=FreeMonoid("a","b");;a:=f.1;;b:=f.2;;
gap> rels:=[[a^3,One(f)],[b^3,One(f)],
           [(a*b)^3,One(f)]];
gap> m:=f/rels;;
gap> k:=KnuthBendixRewritingSystem(m);;
gap> MakeConfluent(k);k;
```

Rewriting System with rules  $a^3 \rightarrow 1, b^3 \rightarrow 1,$   
 $b^2a^2 \rightarrow abab, baba \rightarrow a^2b^2.$

In particular all words  $a^2ba^2ba^2b \dots$  are reduced, thus the group is infinite.

# Example

$$\langle a, b \mid a^3 = b^3 = (ab)^3 = 1 \rangle$$

```
gap> f:=FreeMonoid("a","b");;a:=f.1;;b:=f.2;;
gap> rels:=[[a^3,One(f)],[b^3,One(f)],
           [(a*b)^3,One(f)]];
gap> m:=f/rels;;
gap> k:=KnuthBendixRewritingSystem(m);;
gap> MakeConfluent(k);k;
```

Rewriting System with rules  $a^3 \rightarrow 1$ ,  $b^3 \rightarrow 1$ ,  
 $b^2a^2 \rightarrow abab$ ,  $baba \rightarrow a^2b^2$ .

In particular all words  $a^2ba^2ba^2b \dots$  are reduced, thus the group is infinite.

# Example

$$\langle a, b \mid a^3 = b^3 = (ab)^3 = 1 \rangle$$

```
gap> f:=FreeMonoid("a","b");;a:=f.1;;b:=f.2;;
gap> rels:=[[a^3,One(f)],[b^3,One(f)],
           [(a*b)^3,One(f)]];
gap> m:=f/rels;;
gap> k:=KnuthBendixRewritingSystem(m);;
gap> MakeConfluent(k);k;
```

Rewriting System with rules  $a^3 \rightarrow 1$ ,  $b^3 \rightarrow 1$ ,  
 $b^2a^2 \rightarrow abab$ ,  $baba \rightarrow a^2b^2$ .

In particular all words  $a^2ba^2ba^2b \dots$  are reduced, thus the group is infinite.

# Extensions

or: Cohomology calculations

Now consider the case of  $N \triangleleft G$ . We assume we know  $N$  and  $G/N$ , including rewriting systems  $N = \langle \underline{n} \mid R_1 \rangle$ ,  $G/N = \langle \underline{g} \mid R_2 \rangle$  for both groups.

We want to obtain a rewriting system for  $G$ , respectively describe all possible groups of this kind.

(You probably know the “direct product” and “semidirect product” – there are often more.)

Consider the union of the alphabets  $\underline{g} \cup \underline{n}$  and the following rules:

- $R_1$  as rules on  $\underline{n}$ .
- Every rule  $l(\underline{g}) = r(\underline{g}) \in R_2$  becomes  $l(\underline{g}) = r(\underline{g}) \cdot w(\underline{n})$ , as factor group elements become coset representatives.
- We also get rules  $ng = g\tilde{n}$  describing the action of  $G$  on  $N$ . (Technically we need to use a “Wreath product ordering”, something analogous to an elimination ordering for polynomials.)

# Extensions

or: Cohomology calculations

Now consider the case of  $N \triangleleft G$ . We assume we know  $N$  and  $G/N$ , including rewriting systems  $N = \langle \underline{n} \mid R_1 \rangle$ ,  $G/N = \langle \underline{g} \mid R_2 \rangle$  for both groups.

We want to obtain a rewriting system for  $G$ , respectively describe all possible groups of this kind.

(You probably know the “direct product” and “semidirect product” – there are often more.)

Consider the union of the alphabets  $\underline{g} \cup \underline{n}$  and the following rules:

- $R_1$  as rules on  $\underline{n}$ .
- Every rule  $l(\underline{g}) = r(\underline{g}) \in R_2$  becomes  $l(\underline{g}) = r(\underline{g}) \cdot w(\underline{n})$ , as factor group elements become coset representatives.
- We also get rules  $ng = g\tilde{n}$  describing the action of  $G$  on  $N$ . (Technically we need to use a “Wreath product ordering”, something analogous to an elimination ordering for polynomials.)

# Extensions

or: Cohomology calculations

Now consider the case of  $N \triangleleft G$ . We assume we know  $N$  and  $G/N$ , including rewriting systems  $N = \langle \underline{n} \mid R_1 \rangle$ ,  $G/N = \langle \underline{g} \mid R_2 \rangle$  for both groups.

We want to obtain a rewriting system for  $G$ , respectively describe all possible groups of this kind.

(You probably know the “direct product” and “semidirect product” – there are often more.)

Consider the union of the alphabets  $\underline{g} \cup \underline{n}$  and the following rules:

- $R_1$  as rules on  $\underline{n}$ .
- Every rule  $l(\underline{g}) = r(\underline{g}) \in R_2$  becomes  $l(\underline{g}) = r(\underline{g}) \cdot w(\underline{n})$ , as factor group elements become coset representatives.
- We also get rules  $ng = g\tilde{n}$  describing the action of  $G$  on  $N$ . (Technically we need to use a “Wreath product ordering”, something analogous to an elimination ordering for polynomials.)

# Extensions

or: Cohomology calculations

Now consider the case of  $N \triangleleft G$ . We assume we know  $N$  and  $G/N$ , including rewriting systems  $N = \langle \underline{n} \mid R_1 \rangle$ ,  $G/N = \langle \underline{g} \mid R_2 \rangle$  for both groups.

We want to obtain a rewriting system for  $G$ , respectively describe all possible groups of this kind.

(You probably know the “direct product” and “semidirect product” – there are often more.)

Consider the union of the alphabets  $\underline{g} \cup \underline{n}$  and the following rules:

- $R_1$  as rules on  $\underline{n}$ .
- Every rule  $l(\underline{g}) = r(\underline{g}) \in R_2$  becomes  $l(\underline{g}) = r(\underline{g}) \cdot w(\underline{n})$ , as factor group elements become coset representatives.
- We also get rules  $ng = g\tilde{n}$  describing the action of  $G$  on  $N$ . (Technically we need to use a “Wreath product ordering”, something analogous to an elimination ordering for polynomials.)

# Extensions

or: Cohomology calculations

Now consider the case of  $N \triangleleft G$ . We assume we know  $N$  and  $G/N$ , including rewriting systems  $N = \langle \underline{n} \mid R_1 \rangle$ ,  $G/N = \langle \underline{g} \mid R_2 \rangle$  for both groups.

We want to obtain a rewriting system for  $G$ , respectively describe all possible groups of this kind.

(You probably know the “direct product” and “semidirect product” – there are often more.)

Consider the union of the alphabets  $\underline{g} \cup \underline{n}$  and the following rules:

- $R_1$  as rules on  $\underline{n}$ .
- Every rule  $l(\underline{g}) = r(\underline{g}) \in R_2$  becomes  $l(\underline{g}) = r(\underline{g}) \cdot w(\underline{n})$ , as factor group elements become coset representatives.
- We also get rules  $ng = g\tilde{n}$  describing the action of  $G$  on  $N$ . (Technically we need to use a “Wreath product ordering”, something analogous to an elimination ordering for polynomials.)

# Confluence conditions

The structure described is a group and if and only if the rewriting system constructed this way is confluent.

If we assume that the rewriting system for  $G/N$  is confluent, this imposes conditions in  $N$ .

If  $N$  is elementary abelian these conditions lead to a system on linear equations (the group of 2-cocycles for this  $G/N$  module  $N$ ).

We want to use this idea to find larger quotients for a finitely presented group.

# Confluence conditions

The structure described is a group and if and only if the rewriting system constructed this way is confluent.

If we assume that the rewriting system for  $G/N$  is confluent, this imposes conditions in  $N$ .

If  $N$  is elementary abelian these conditions lead to a system on linear equations (the group of 2-cocycles for this  $G/N$  module  $N$ ).

We want to use this idea to find larger quotients for a finitely presented group.

# Confluence conditions

The structure described is a group and if and only if the rewriting system constructed this way is confluent.

If we assume that the rewriting system for  $G/N$  is confluent, this imposes conditions in  $N$ .

If  $N$  is elementary abelian these conditions lead to a system on linear equations (the group of 2-cocycles for this  $G/N$  module  $N$ ).

We want to use this idea to find larger quotients for a finitely presented group.

# Quotient Algorithms

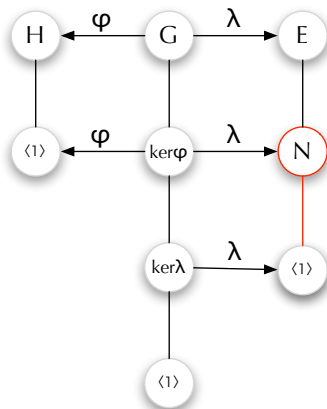
Suppose  $G$  is finitely presented and we know an epimorphism  $\varphi: G \rightarrow H$  for a finite group  $H$ .

We want to find a new epimorphism (a "lift")  $\lambda: G \rightarrow E$ , such that  $\ker \lambda < \ker \varphi$ .

This means that

$$N := (\ker \varphi)^\lambda \triangleleft E$$

with  $E/N \cong H$ .



# Assumptions

We shall assume  $N \cong \ker \varphi / \ker \lambda$  is elementary abelian. This way we work over a prime field.

By iteration we can construct a solvable  $N$ .

Such algorithms have been investigated extensively for various classes of groups:  $p$ -quotient, nilpotent quotient, solvable quotient.

Want to generalize to nonsolvable quotients. Only assume confluent rewriting for the known quotient  $H$  (joint work with A. NIEMEYER, Perth, WA).

We could in principle find the largest possible  $N$  by rewriting a presentation for  $\ker \varphi$  and abelianizing this presentation – due to the growing index this very quickly becomes infeasible.

# Assumptions

We shall assume  $N \cong \ker \varphi / \ker \lambda$  is elementary abelian. This way we work over a prime field.

By iteration we can construct a solvable  $N$ .

Such algorithms have been investigated extensively for various classes of groups:  $p$ -quotient, nilpotent quotient, solvable quotient.

Want to generalize to nonsolvable quotients. Only assume confluent rewriting for the known quotient  $H$  (joint work with A. NIEMEYER, Perth, WA).

We could in principle find the largest possible  $N$  by rewriting a presentation for  $\ker \varphi$  and abelianizing this presentation – due to the growing index this very quickly becomes infeasible.

# Assumptions

We shall assume  $N \cong \ker \varphi / \ker \lambda$  is elementary abelian. This way we work over a prime field.

By iteration we can construct a solvable  $N$ .

Such algorithms have been investigated extensively for various classes of groups:  $p$ -quotient, nilpotent quotient, solvable quotient.

Want to generalize to nonsolvable quotients. Only assume confluent rewriting for the known quotient  $H$  (joint work with A. NIEMEYER, Perth, WA).

We could in principle find the largest possible  $N$  by rewriting a presentation for  $\ker \varphi$  and abelianizing this presentation – due to the growing index this very quickly becomes infeasible.

# Assumptions

We shall assume  $N \cong \ker \varphi / \ker \lambda$  is elementary abelian. This way we work over a prime field.

By iteration we can construct a solvable  $N$ .

Such algorithms have been investigated extensively for various classes of groups:  $p$ -quotient, nilpotent quotient, solvable quotient.

Want to generalize to nonsolvable quotients. Only assume confluent rewriting for the known quotient  $H$  (joint work with A. NIEMEYER, Perth, WA).

We could in principle find the largest possible  $N$  by rewriting a presentation for  $\ker \varphi$  and abelianizing this presentation – due to the growing index this very quickly becomes infeasible.

# Building a larger quotient

Instead we consider (for a given prime  $p$ ) a free module  $M$ , generated by the cofactors  $w(\underline{n}) = r(\underline{g})^{-1} \cdot l(\underline{g})$ .  $N$  will be a quotient of  $M$ .

We then consider the overlaps of left hand sides.

Only rules "lifted from"  $H \cong E/N$  are interesting. As the rules hold in  $H$  we get equations in  $M$ .

# Building a larger quotient

Instead we consider (for a given prime  $p$ ) a free module  $M$ , generated by the cofactors  $w(\underline{n}) = r(\underline{g})^{-1} \cdot l(\underline{g})$ .  $N$  will be a quotient of  $M$ .

We then consider the overlaps of left hand sides.

Only rules "lifted from"  $H \cong E/N$  are interesting. As the rules hold in  $H$  we get equations in  $M$ .

# Module presentation

Next we evaluate relators for  $G$  in  $E$ . Because they are trivial in  $H$ , they yield further equations in  $M$ .

(There are some messy technicalities about how to define these images to get an epimorphism onto  $E$ .)

The resulting equations give a **module presentation** for  $N$ : They are linear equations in free generators of  $M$  under elements of the acting group algebra  $\mathbb{F}_p H$ .

If  $N$  is central ( $p$ -quotient algorithm), there is no action, and we have just linear equations.

# Module presentation

Next we evaluate relators for  $G$  in  $E$ . Because they are trivial in  $H$ , they yield further equations in  $M$ .

(There are some messy technicalities about how to define these images to get an epimorphism onto  $E$ .)

The resulting equations give a **module presentation** for  $N$ : They are linear equations in free generators of  $M$  under elements of the acting group algebra  $\mathbb{F}_p H$ .

If  $N$  is central ( $p$ -quotient algorithm), there is no action, and we have just linear equations.

# Module presentation

Next we evaluate relators for  $G$  in  $E$ . Because they are trivial in  $H$ , they yield further equations in  $M$ .

(There are some messy technicalities about how to define these images to get an epimorphism onto  $E$ .)

The resulting equations give a **module presentation** for  $N$ : They are linear equations in free generators of  $M$  under elements of the acting group algebra  $\mathbb{F}_p H$ .

If  $N$  is central ( $p$ -quotient algorithm), there is no action, and we have just linear equations.

# Module presentation

**Example:**  $G = S_4$ ,  $H = S_3 = \langle x_1, x_2 \rangle$ ,  $p = 2$ . Concrete calculation yields a free  $\mathbb{F}_2 S_3$ -module  $\langle z_1, \dots, z_6 \rangle$  of rank 5 and relations:

$$\begin{array}{lll} z_1^{x_1-1} & z_2^{x_2-1} & z_1 z_2^{-x_1 x_2} z_3^{x_1} \\ z_1^{x_2 x_1} z_2^{-1} z_3^{1+x_2} & z_1^{x_2-x_2 x_1} z_2^{1-x_1} z_3^{x_1 x_2-1} & z_1^{1+x_2 x_1} z_2^{x_1} z_4^{1+x_1 x_2 x_1} \\ z_1 z_5^{1+x_1} & z_1^{1+x_2 x_1} z_2^{x_1} z_6^{1+x_1 x_2 x_1} & z_1^{1+x_1+2x_2 x_1} z_2^{2x_1} z_4^{2x_1 x_2 x_1} z_6^2 \\ & \vdots & \end{array}$$

(full example in preprint at  
<http://www.math.colostate.edu/~hulpke/hq>)

# Making the module concrete

The task now is to find a basis for the quotient module defined by these relators as well as matrices for the action of  $H$  on it. Together this permits us to construct the new, larger quotient group.

The following methods have been used:

Vector Enumeration A process similar to the Todd-Coxeter coset enumeration. It enumerates vector images under the algebra and tries to deduce dependence relations.

Testing irreducible modules (PLESKENS version of the SQ does this implicitly by cohomology calculations) Run through all irreducible  $\mathbb{F}_p H$ -modules and check which modules fulfill the relations.

Noncommutative Gröbner bases ? Ideas for efficient solution are welcome!

# Making the module concrete

The task now is to find a basis for the quotient module defined by these relators as well as matrices for the action of  $H$  on it. Together this permits us to construct the new, larger quotient group.

The following methods have been used:

**Vector Enumeration** A process similar to the Todd-Coxeter coset enumeration. It enumerates vector images under the algebra and tries to deduce dependence relations.

Testing irreducible modules (PLESKENS version of the SQ does this implicitly by cohomology calculations) Run through all irreducible  $\mathbb{F}_p H$ -modules and check which modules fulfill the relations.

Noncommutative Gröbner bases ? Ideas for efficient solution are welcome!

# Making the module concrete

The task now is to find a basis for the quotient module defined by these relators as well as matrices for the action of  $H$  on it. Together this permits us to construct the new, larger quotient group.

The following methods have been used:

**Vector Enumeration** A process similar to the Todd-Coxeter coset enumeration. It enumerates vector images under the algebra and tries to deduce dependence relations.

**Testing irreducible modules** (PLESKENS version of the SQ does this implicitly by cohomology calculations) Run through all irreducible  $\mathbb{F}_p H$ -modules and check which modules fulfill the relations.

Noncommutative Gröbner bases ? Ideas for efficient solution are welcome!

# Making the module concrete

The task now is to find a basis for the quotient module defined by these relators as well as matrices for the action of  $H$  on it. Together this permits us to construct the new, larger quotient group.

The following methods have been used:

**Vector Enumeration** A process similar to the Todd-Coxeter coset enumeration. It enumerates vector images under the algebra and tries to deduce dependence relations.

**Testing irreducible modules** (PLESKENS version of the SQ does this implicitly by cohomology calculations) Run through all irreducible  $\mathbb{F}_p H$ -modules and check which modules fulfill the relations.

**Noncommutative Gröbner bases** ? Ideas for efficient solution are welcome!

# Initialization

To start this algorithm we can use the largest abelian quotient for  $p$ -quotients or solvable quotients.

Otherwise we can use any quotient found – for example permutation group quotients (found via a combinatorial search) or matrix group quotients as described above.

I have an Implementation of a single step in GAP. Determining a module basis by far dominates the algorithm.

# Initialization

To start this algorithm we can use the largest abelian quotient for  $p$ -quotients or solvable quotients.

Otherwise we can use any quotient found – for example permutation group quotients (found via a combinatorial search) or matrix group quotients as described above.

I have an Implementation of a single step in GAP. Determining a module basis by far dominates the algorithm.

# Initialization

To start this algorithm we can use the largest abelian quotient for  $p$ -quotients or solvable quotients.

Otherwise we can use any quotient found – for example permutation group quotients (found via a combinatorial search) or matrix group quotients as described above.

I have an Implementation of a single step in GAP. Determining a module basis by far dominates the algorithm.



# Examples

Group	$\text{Img}\varphi$	Module	Time
$G_1 = \langle x, y, z \mid [x, [y, z]] = z, [y, [y, z]] = x, [z, [z, x]] = y \rangle$	$A_5$	$2^5$	4
$G_2 = \langle x, y, z \mid x^2 = y^5 = z^3 = (xy)^3 = (xz)^3 = (y^2z)^2 = 1 \rangle$	$A_5$	$2^5$	1
	$A_6$	$2^{14}$ $3^{11}$	1072 1191

# More Examples

$$\begin{aligned}G_3 &= \langle x, y, z \mid x^2 = y^5 = z^3 \\ &= (xy)^3 = (xz)^2 \\ &= (y^2z)^3 = 1 \rangle\end{aligned}$$

 $A_6$  $2^{20}$ 

959

---

$$\begin{aligned}G_4 &= \langle a, b \mid aba^{-2}bab^{-1} \\ &= (b^{-1}a^3b^{-1}a^{-3})^2a = 1 \rangle\end{aligned}$$

 $L_2(11)$  $2^{53}$ 

96367

 $3^{52}$ 

226533

---

$$\begin{aligned}G_5 &= \langle a, b \mid a^2 = b^3 \\ &= (ab)^7 = 1 \rangle\end{aligned}$$

 $L_2(7)$  $2^6$ 

375

# Examples Galore

$$G_6 = \langle a, b \mid ba^{-2}ba^{-1}b^2ab^2a^{-1} \\ = abab^2aba^2b^{-1}a = 1 \rangle$$

$L_2(7)$      $2^1$     1057

$3^{22}$     690

$7^3$     780

---

$$G_7 = \langle a, b, c \mid ac^{-1}bc^{-1}aba^{-1}b \\ = abab^{-1}c^2b^{-1} \\ = a^2b^{-1}(ca)^4cb^{-1} = 1 \rangle$$

$SL_2(5)$      $5^4$     423

$11^4$     494