The OSCAR Computer Algebra System

Introduction and Examples

Lars Göttgens, RWTH Aachen University

. . . and the OSCAR Development Team

https://oscar-system.org/

September 5, 2025

What is OSCAR?
https://oscar-system.org/

- Open Source Computer Algebra Research system
- A tool for interdisciplinary research and computations in algebra, geometry, and number theory
- Also a toolkit for applications of algebra (coding theory, cryptography, bioinformatics, ...)
- Funded by German Research Council (DFG) from 2017-2028
- Contributions from 118 individuals from all over the world
- Open to contributions from anyone, curated via code reviews
- All parts of OSCAR are open source

OSCAR Is Both Old and New

- OSCAR is “new”...

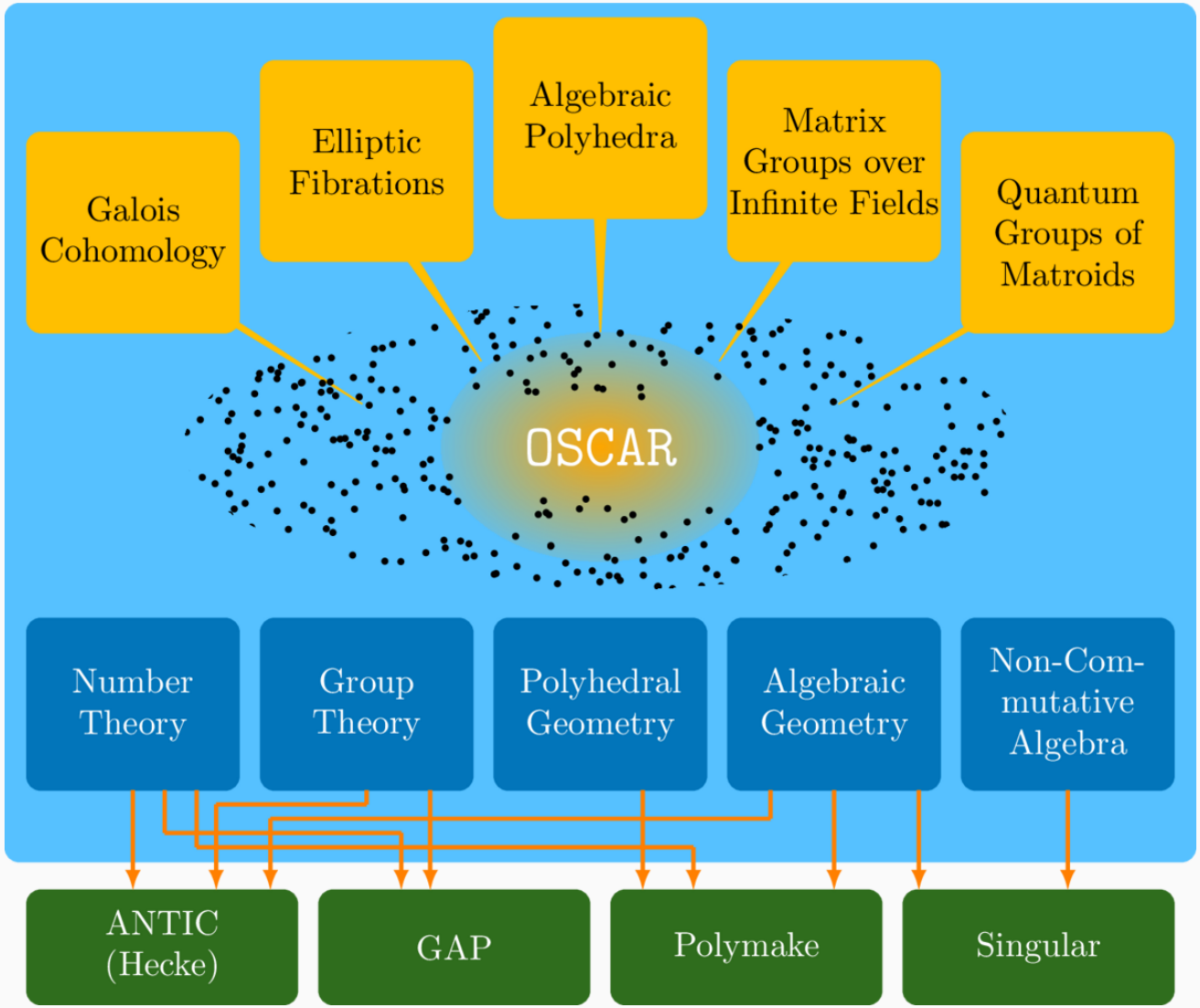
OSCAR Is Both Old and New

- OSCAR is “new”...
- ...but based on pre-existing systems, the four *cornerstones*:
  - ANTIC (Nemo, Hecke)
  - GAP
  - polymake
  - Singular

OSCAR Is Both Old and New

- OSCAR is "new"...
- ...but based on pre-existing systems, the four *cornerstones*:
  - ANTIC (Nemo, Hecke)
  - GAP
  - polymake
  - Singular
- constituting ≥ 30 years of work and a few million lines of code
- tied together and extended by code written in the Julia programming language
- development of OSCAR and cornerstones is tightly intertwined
- cornerstone creators & custodians are OSCAR team members

The structure of OSCAR



OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...
- number theory: class groups, Galois groups, ...

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...
- number theory: class groups, Galois groups, ...
- algebraic geometry: schemes, (elliptic) curves, toric varieties, ...

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...
- number theory: class groups, Galois groups, ...
- algebraic geometry: schemes, (elliptic) curves, toric varieties, ...
- polyhedral geometry, tropical geometry

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...
- number theory: class groups, Galois groups, ...
- algebraic geometry: schemes, (elliptic) curves, toric varieties, ...
- polyhedral geometry, tropical geometry
- noncommutative algebra: PBW-algebras, GR-algebras, ...

OSCAR Is More Than Group Theory

- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...
- number theory: class groups, Galois groups, ...
- algebraic geometry: schemes, (elliptic) curves, toric varieties, ...
- polyhedral geometry, tropical geometry
- noncommutative algebra: PBW-algebras, GR-algebras, ...
- Lie theory: root systems, Weyl groups, Lie algebras, modules, ...

OSCAR Is More Than Group Theory

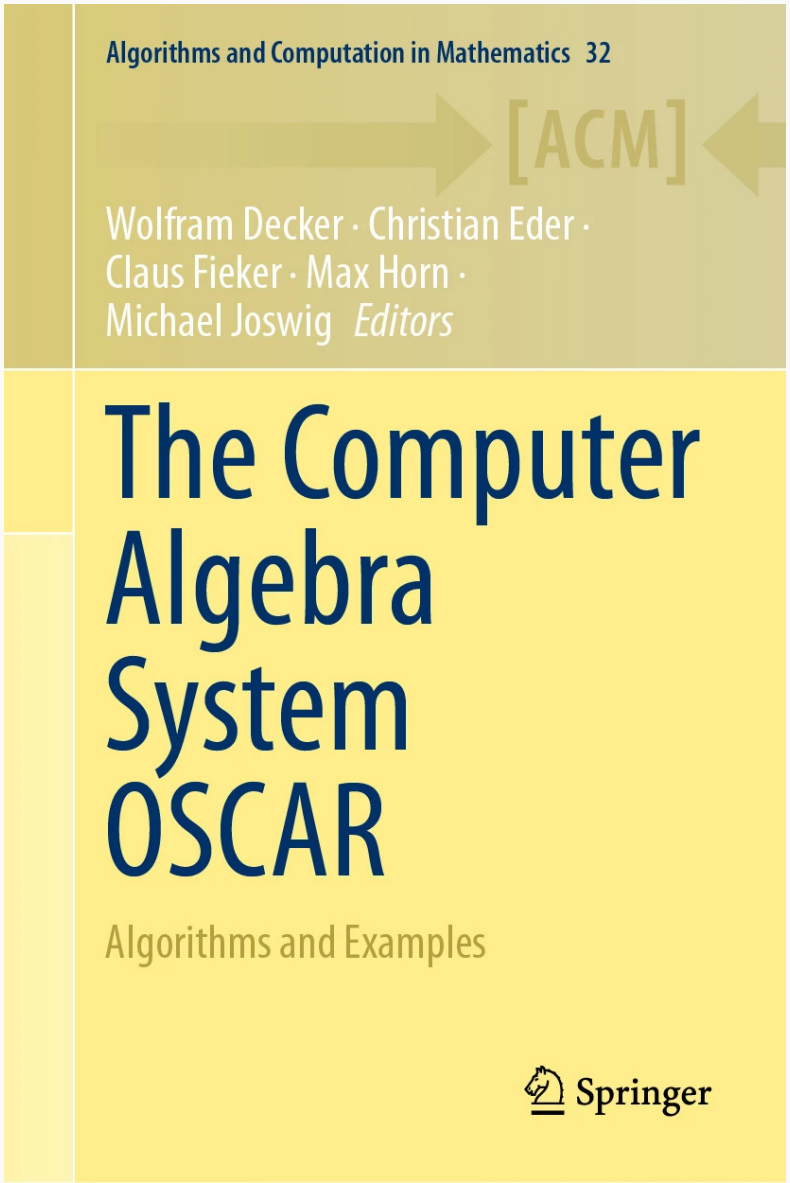
- *efficient* basic arithmetic: polynomials, matrices, finite fields, number fields, power series, groups, ...with common interfaces
- generic and specialized optimized linear algebra
- group theory: permutation groups, finitely presented groups, matrix groups, group cohomology, ...
- commutative algebra: Gröbner bases, (graded) modules over fin. gen. rings, affine algebras, primary decomposition, ...
- number theory: class groups, Galois groups, ...
- algebraic geometry: schemes, (elliptic) curves, toric varieties, ...
- polyhedral geometry, tropical geometry
- noncommutative algebra: PBW-algebras, GR-algebras, ...
- Lie theory: root systems, Weyl groups, Lie algebras, modules, ...
- ...and much more is there, or to come

The OSCAR Book

Get the OSCAR book for many worked out examples

19 chapters with topics ranging from basics to current research.

See book.oscar-system.org



Why Julia?

Performance

- Solves the 2-language problem:
  - easy to write
  - near C performance due to JIT compilation
- supports parallel computing

Why Julia?

Performance

- Solves the 2-language problem:
  - easy to write
  - near C performance due to JIT compilation
- supports parallel computing

Community and Ecosystem

- Open Source (MIT License)
- supports multiple platforms
- designed by mathematically minded people
- large, growing ecosystem

Why Julia?

Julia features

- easy C interoperability; good C++ support
- good support for interactive use in the REPL:
  - searchable history, tab-completion, help mode
- first-class Jupyter notebook support
- dynamically typed
- automatic memory management (garbage collector)

Examples

Topics:

1. Algebraic Lie Theory: Root systems, Weyl groups, Lie algebras, monomial bases
2. Group Theory: an overview of different group types

Introduction to Group Theory

in OSCAR

Lars Göttgens, RWTH Aachen University

Examples by C. Fieker and M. Horn

https://oscar-system.org/

September 5, 2025

Feature overview

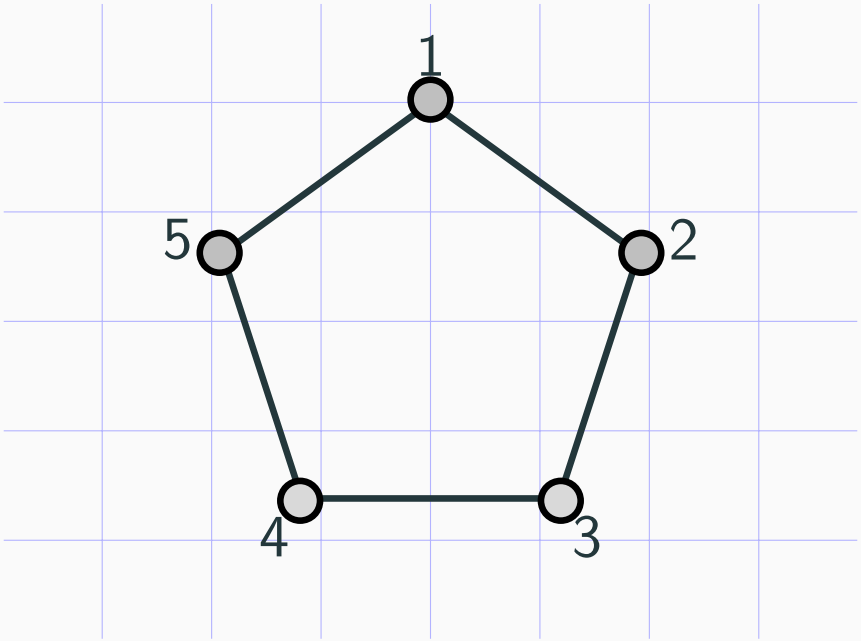
- Different types of groups,
- subgroups, quotients,
- products,
- G-sets,
- group libraries,
- character theory,
- G-modules,
- ...

Feature overview (this talk)

- **Different types of groups,**
- **subgroups, quotients,**
- products,
- G-sets,
- group libraries,
- character theory,
- G-modules,
- ...

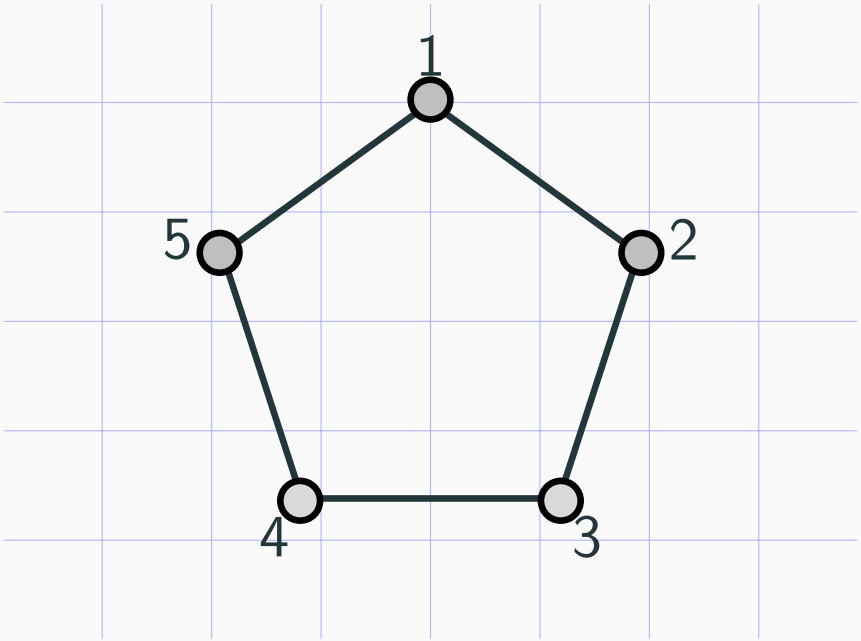
Example: Dihedral group D_{10}

Consider a regular pentagon:



Example: Dihedral group D_{10}

Consider a regular pentagon:

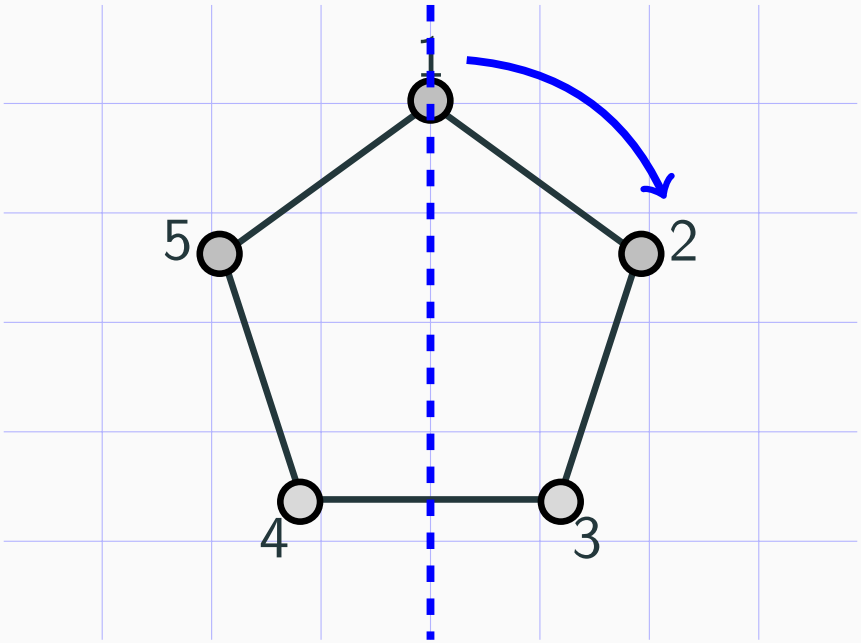


It has the following symmetries:

- reflections s_i at symmetry axes through vertex i
- rotations by multiples of 72 degrees

Example: Dihedral group D_{10}

Consider a regular pentagon:



It has the following symmetries:

- reflections s_i at symmetry axes through vertex i
- rotations by multiples of 72 degrees

D_{10} is generated by $s := s_1$ and a 72 degree rotation r.

Permutation groups

Consider the action of the symmetries on the vertices:

- s leaves vertex 1 invariant, swaps 2 with 5, and 3 with 4.
- r maps i to $i+1$, and 5 to 1.

```
julia> G_perm = @permutation_group(5, (2,5)(3,4), (1,2,3,4,5))
Permutation group of degree 5
```

or

```
julia> G_perm, emb = sub(symmetric_group(5),
↪  [cperm([2,5],[3,4]), cperm([1,2,3,4,5])])
(Permutation group of degree 5, Hom: G_sub -> Sym(5))
```

Permutation groups

Consider the action of the symmetries on the vertices:

- s leaves vertex 1 invariant, swaps 2 with 5, and 3 with 4.
- r maps i to $i+1$, and 5 to 1.

```
julia> G_perm = @permutation_group(5, (2,5)(3,4), (1,2,3,4,5))
Permutation group of degree 5
```

or

```
julia> G_perm, emb = sub(symmetric_group(5),
↪  [cperm([2,5],[3,4]), cperm([1,2,3,4,5])])
(Permutation group of degree 5, Hom: G_sub -> Sym(5))
```

Verify order and iso type:

```
julia> order(G_perm)
10

julia> describe(G_perm)
"D10"
```

Permutation group elements

Conjugating a rotation by a reflection changes the direction:

```
julia> s, r = gens(G_perm);

julia> s^-1 * r * s
(1,5,4,3,2)
```

Permutation group elements

Conjugating a rotation by a reflection changes the direction:

```
julia> s, r = gens(G_perm);

julia> s^-1 * r * s
(1,5,4,3,2)
```

There are several ways to enter permutations:

```
julia> g = perm([1,2,3,5,4]) # 'tabular' notation
(4,5)
julia> g in G_perm
false

julia> G_perm([2,1,5,4,3]) # 'tabular' notation with explicit parent
(1,2)(3,5)

julia> cperm([1,2],[3,5]) # cycle decomposition
(1,2)(3,5)

julia> @perm (1,2)(3,5)
(1,2)(3,5)
```

Matrix groups

Embed the pentagon into the real plane $\mathbb{R}^2$ and describe automorphisms as linear transformations:

- s is a reflection in the y-axis,
- r rotates by 72 degree, i.e. $\frac{2\pi}{5}$ radians.

```
julia> K = algebraic_closure(QQ);

julia> mat_s = matrix(K, [ -1 0 ; 0 1 ]);

julia> s, c = sinpi(2*one(K)/5), cospi(2*one(K)/5)
(Root 0.951057 of 16x^4 - 20x^2 + 5, Root 0.309017 of 4x^2 + 2x - 1)

julia> mat_r = matrix(K, [ c -s ; s c ]);

julia> G_mat = matrix_group(mat_s, mat_r)
Matrix group of degree 2
  over algebraic closure of rational field
```

Matrix groups

Embed the pentagon into the real plane $\mathbb{R}^2$ and describe automorphisms as linear transformations:

- s is a reflection in the y-axis,
- r rotates by 72 degree, i.e. $\frac{2\pi}{5}$ radians.

```
julia> K = algebraic_closure(QQ);

julia> mat_s = matrix(K, [ -1 0 ; 0 1 ]);

julia> s, c = sinpi(2*one(K)/5), cospi(2*one(K)/5)
(Root 0.951057 of 16x^4 - 20x^2 + 5, Root 0.309017 of 4x^2 + 2x - 1)

julia> mat_r = matrix(K, [ c -s ; s c ]);

julia> G_mat = matrix_group(mat_s, mat_r)
Matrix group of degree 2
  over algebraic closure of rational field
```

```
julia> is_isomorphic(G_mat, G_perm)
true
```

Actually construct the pentagon

Vertex 1 has coordinates $(0, 1)$. We get the other ones via the group action

```
julia> p = [K(0), K(1)]; # coordinates of vertex 1

julia> orb = orbit(G_mat, *, p)
G-set of
  matrix group of degree 2 over QQBar
  with seeds [[Root 0 of x, Root 1.00000 of x - 1]]
```

Actually construct the pentagon

Vertex 1 has coordinates $(0, 1)$. We get the other ones via the group action

```
julia> p = [K(0), K(1)]; # coordinates of vertex 1

julia> orb = orbit(G_mat, *, p)
G-set of
  matrix group of degree 2 over QQBar
  with seeds [[Root 0 of x, Root 1.00000 of x - 1]]
```

We obtain the orbit as a lazy G-set. Actual computations only happen when asking for the elements:

```
julia> pts = collect(orb)
5-element Vector{Vector{QQBarFieldElem}}:
 [Root 0 of x, Root 1.00000 of x - 1]
 [Root 0.951057 of 16x^4 - 20x^2 + 5, Root 0.309017 of 4x^2 + 2x - 1]
 [Root -0.951057 of 16x^4 - 20x^2 + 5, Root 0.309017 of 4x^2 + 2x - 1
 [Root 0.587785 of 16x^4 - 20x^2 + 5, Root -0.809017 of 4x^2 + 2x - 1
 [Root -0.587785 of 16x^4 - 20x^2 + 5, Root -0.809017 of 4x^2 + 2x -

julia> pts = pts[[1,2,4,5,3]]; # permute to match previous slides
```

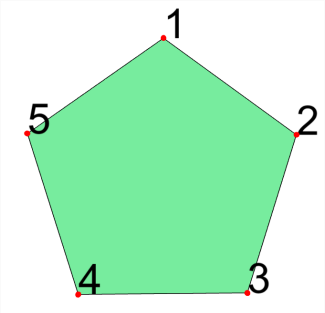
Actually construct the pentagon (2)

```
julia> P = convex_hull(pts)
Polyhedron in ambient dimension 2 with QQBarFieldElem type
↪  coefficients
```

Actually construct the pentagon (2)

```
julia> P = convex_hull(pts)
Polyhedron in ambient dimension 2 with QQBarFieldElem type
↪  coefficients
```

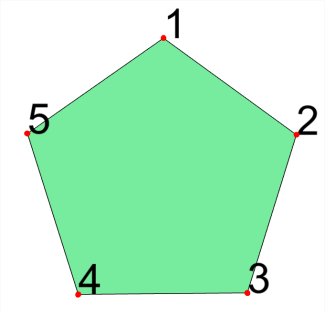
```
julia> visualize(P)
```



Actually construct the pentagon (2)

```
julia> P = convex_hull(pts)
Polyhedron in ambient dimension 2 with QQBarFieldElem type
↪  coefficients
```

```
julia> visualize(P)
```



And back to groups:

```
julia> G_aut = automorphism_group(P; type=:linear,
↪  action=:on_vertices)
Permutation group of degree 5

julia> describe(G_aut)
"D10"
```

Generic groups

Defined by arbitrary objects as group elements or generators, and a binary function.

```
julia> V = vector_space(K, 2)

julia> hom_s = hom(V, V, [-gen(V,1), gen(V,2)]);

julia> hom_r = hom(V, V, mat_r);

julia> G_generic = generic_group(closure([hom_s, hom_r], *), *)[1]
Generic group of order 10 with multiplication table
```

Generic groups

Defined by arbitrary objects as group elements or generators, and a binary function.

```
julia> V = vector_space(K, 2)

julia> hom_s = hom(V, V, [-gen(V,1), gen(V,2)]);

julia> hom_r = hom(V, V, mat_r);

julia> G_generic = generic_group(closure([hom_s, hom_r], *), *)[1]
Generic group of order 10 with multiplication table
```

Generic groups use a multiplication table, so are only suitable for small examples.
$\rightarrow$ still good for intuition and teaching

Generic groups

Defined by arbitrary objects as group elements or generators, and a binary function.

```
julia> V = vector_space(K, 2)

julia> hom_s = hom(V, V, [-gen(V,1), gen(V,2)]);

julia> hom_r = hom(V, V, mat_r);

julia> G_generic = generic_group(closure([hom_s, hom_r], *), *)[1]
Generic group of order 10 with multiplication table
```

Generic groups use a multiplication table, so are only suitable for small examples.

$\rightarrow$ still good for intuition and teaching

And we can convert

```
julia> iso = isomorphism(PermGroup, G_generic);

julia> G_generic_perm = codomain(iso)
Permutation group of degree 10
```

Finitely presented groups

Described by generators and relations.

Construction from an already existing group:

```
julia> G_fp = fp_group(G_perm)
Finitely presented group of order 10

julia> gens(G_fp)
2-element Vector{FPGroupElem}:
 F1
 F2

julia> relators(G_fp)
3-element Vector{FPGroupElem}:
 F1^2
 F1^-1*F2*F1*F2^-4
 F2^5
```

Finitely presented groups: construction

$$D_{10} = \langle s, r \mid s^2 = r^5 = (sr)^2 = 1 \rangle$$

Construction as a quotient of a free group:

```
julia> F = free_group(2)
Free group of rank 2

julia> G_fp, proj = quo(F, [F[1]^2, F[2]^5, (F[1]*F[2])^2])
(Finitely presented group, Hom: F -> G_fp)

julia> describe(G_fp)
"D10"

julia> is_isomorphic(G_fp, G_perm)
true
```

Dedicated constructors

For many "natural occurring groups", there are dedicated constructors:

Construction as a quotient of a free group:

```
julia> dihedral_group(10)
Pc group of order 10

julia> dihedral_group(PermGroup, 10)
Permutation group of degree 5
```

Connection to GAP

- Everything from GAP is available in OSCAR

Connection to GAP

- Everything from GAP is available in OSCAR
- Much of the group theory in OSCAR uses GAP underneath

Connection to GAP

- Everything from GAP is available in OSCAR
- Much of the group theory in OSCAR uses GAP underneath
- OSCAR allows for easy composition of group theory with other areas of OSCAR, e.g.:
  - Matrix groups over number fields
  - Invariant theory of permutation and matrix groups:
     W. Decker, L. Ramesh, and J. Schmitt. "Invariant theory". In: *The computer algebra system OSCAR—algorithms and examples*. Vol. 32. Algorithms Comput. Math. Springer, Cham, 2025, pp. 297–331
  - Using polyhedral geometry for character theory:
     T. Breuer, M. Joswig, and G. Malle. *Zeros of S-characters*. 2025. arXiv: 2408.16785 [math.GR]

Give it a try yourself!

There is a lot more functionality that does not all fit in this talk.

Have a look at the documentation, talk to us, ...

Try it yourself!

On the website you can find:

- installation instructions
- tutorials
- documentation
- links to slack and github

oscar-system.org

Try it yourself!

On the website you can find:

- installation instructions
- tutorials
- documentation
- links to slack and github

oscar-system.org

We are looking forward to your comments, bug reports, feature requests, code contributions, ...

Try it yourself!

On the website you can find:

- installation instructions
- tutorials
- documentation
- links to slack and github

oscar-system.org

We are looking forward to your comments, bug reports, feature requests, code contributions, ...

oscar-system.org/newsletter

Thank you!