The Ideas behind the `homalg` Project

Mohamed Barakat & Markus Lange-Hegermann

TU-Kaiserslautern & RWTH-Aachen

Sage days 23.5,
ITWM & TU-Kaiserslautern,
14-16.07.2010

Computability of ABELian Categories

Definition

We call an ABELian category **computable** if all existential quantifiers appearing in the axioms can be turned into constructive ones.

Example axiom: For any morphism there *exists* a kernel.

The `homalg` Project

The idea: A homological algebra meta-package for ABELian categories

`homalg`

The `homalg` Project

The category of finitely presented modules as the basic example of an ABELian category

$$\begin{tikzcd} & \texttt{homalg} \\ \texttt{Modules} \arrow[ur] & \end{tikzcd}$$

Computable Rings

- A ring R is called **computable**, if there exist algorithms for *solving linear systems*[1] over R.

[1] $\{x \in R^{n\times 1} \,|\, Ax = b\}$ and $\{x \in R^{1\times m} \,|\, xA = c\}$ for $A \in R^{m\times n}, b \in R^{m\times 1}, c \in R^{1\times n}$

Computable Rings

- A ring R is called **computable**, if there exist algorithms for *solving linear systems*[1] over R.
- $K[x_1, \dots, x_n]$ is computable (BUCHBERGER's Algorithm).
  $K[x_1, \dots, x_n]_{\langle x_1, \dots, x_n \rangle}$ is computable (MORA's Algorithm).
  And probably *your* favorite ring is computable.

[1] $\{x \in R^{n \times 1} \mid Ax = b\}$ and $\{x \in R^{1 \times m} \mid xA = c\}$ for $A \in R^{m \times n}, b \in R^{m \times 1}, c \in R^{1 \times n}$

Computable Rings

- A ring R is called **computable**, if there exist algorithms for *solving linear systems*[1] over R.
- $K[x_1, \ldots, x_n]$ is computable (BUCHBERGER's Algorithm).
  $K[x_1, \ldots, x_n]_{\langle x_1, \ldots, x_n \rangle}$ is computable (MORA's Algorithm).
  And probably *your* favorite ring is computable.

[BLH, Theorem 3.4]

Let R be computable. Then the ABELian category of finitely presented R-modules is computable.

"Proof": Use Matrices.

($\square$)

[1] $\{x \in R^{n\times 1} \mid Ax = b\}$ and $\{x \in R^{1\times m} \mid xA = c\}$ for $A \in R^{m\times n}, b \in R^{m\times 1}, c \in R^{1\times n}$

The homalg Project

The category of finitely presented modules as the basic example of an ABELian category

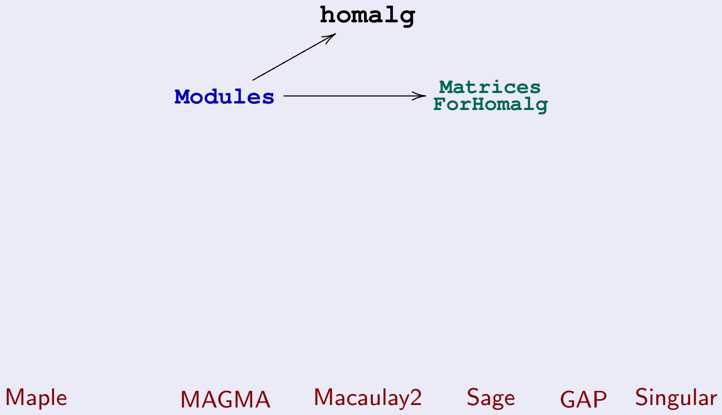
$$\begin{tikzcd} & \texttt{homalg} \\ \texttt{Modules} \arrow[ur] & \end{tikzcd}$$

The `homalg` Project

Matrices provide the needed data structure for finitely presented modules and their morphisms

$$
\begin{tikzcd}
 & \texttt{homalg} & \\
\texttt{Modules} \arrow[ur] \arrow[rr] & & \texttt{MatricesForHomalg}
\end{tikzcd}
$$

Candidates: There are several systems that could host homalg



The `homalg` Project

Candidates: There are several systems that could host `homalg`, each supporting certain kinds of rings

$$\begin{tikzcd} & \texttt{homalg} \\ \texttt{Modules} \arrow[ur] \arrow[r] & \texttt{MatricesForHomalg} \end{tikzcd}$$

Maple	MAGMA	Macaulay2	Sage	GAP	Singular
$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
$\mathbb{Z}[x,\partial]$, $\dots$	$\mathbb{Z}[x]$, $\mathbb{F}[x]$	$\mathbb{F}[x]$, $\mathbb{F}[x,\partial]$, $\dots$	$\mathbb{Z}$, $\dots$	$\mathbb{Z}$	$\mathbb{F}[x]$, $\mathbb{F}[x,\partial]$, $\dots$

The `homalg` Project

GAP4: The best suited *language* for abstract mathematical programming

homalg GAP4

Modules $\longrightarrow$ **MatricesForHomalg**

Maple	MAGMA	Macaulay2	Sage	GAP	Singular
$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
$\mathbb{Z}[x,\partial]$, …	$\mathbb{Z}[x]$, $\mathbb{F}[x]$	$\mathbb{F}[x]$, $\mathbb{F}[x,\partial]$, …	$\mathbb{Z}$, …	$\mathbb{Z}$	$\mathbb{F}[x]$, $\mathbb{F}[x,\partial]$, …

The `homalg` Project

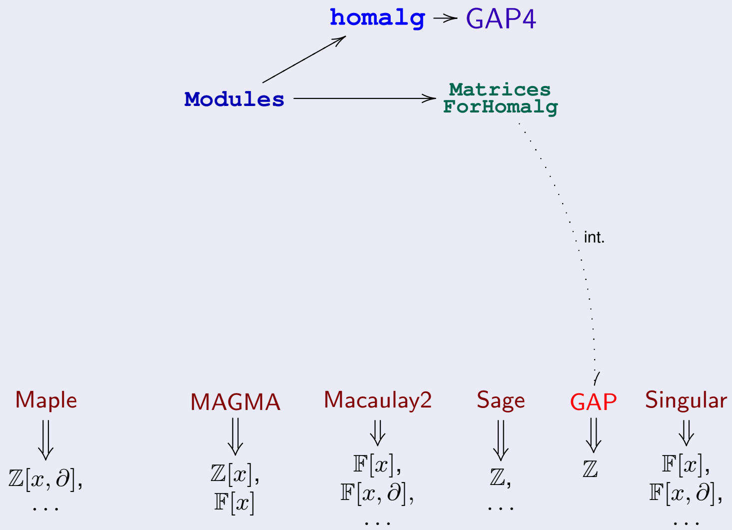
`homalg`: As a GAP4 package

$$\begin{array}{ccc} & \texttt{homalg} \to \text{GAP4} & \\ \nearrow & & \\ \texttt{Modules} & \longrightarrow & \texttt{MatricesForHomalg} \end{array}$$

Maple	MAGMA	Macaulay2	Sage	GAP	Singular
$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
$\mathbb{Z}[x,\partial]$, ...	$\mathbb{Z}[x]$, $\mathbb{F}[x]$	$\mathbb{F}[x]$, $\mathbb{F}[x,\partial]$, ...	$\mathbb{Z}$, ...	$\mathbb{Z}$	$\mathbb{F}[x]$, $\mathbb{F}[x,\partial]$, ...

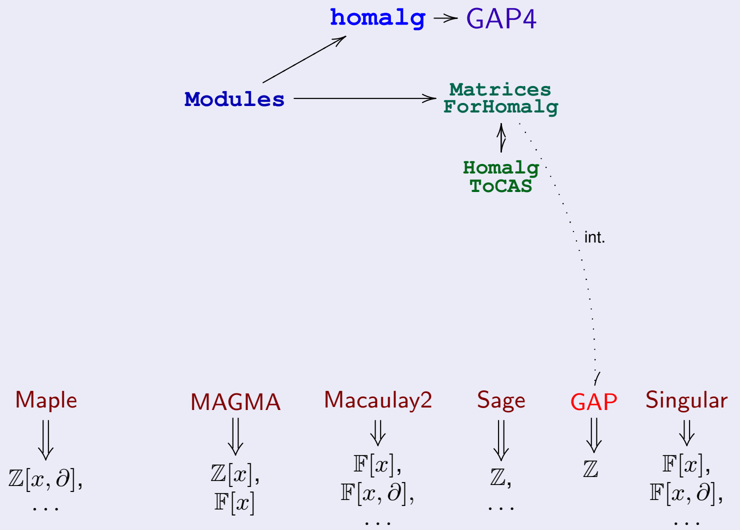
The homalg Project

homalg: GAP “sufficiently supports” the ring of integers



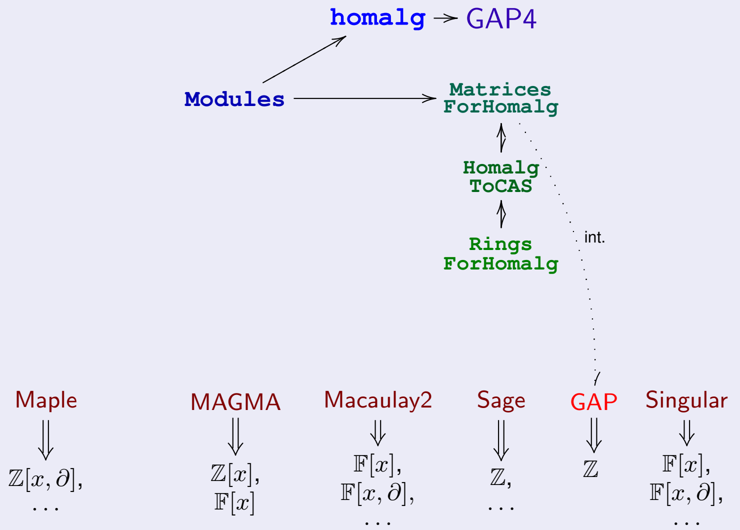
The homalg Project

HomalgToCAS: External objects and the GAP4-representations: external rings and external matrices



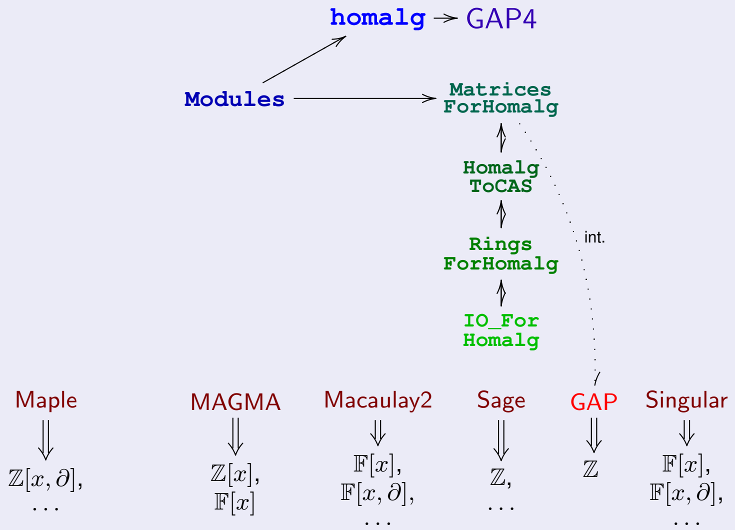
The homalg Project

RingsForHomalg: The dictionaries used by MatricesForHomalg



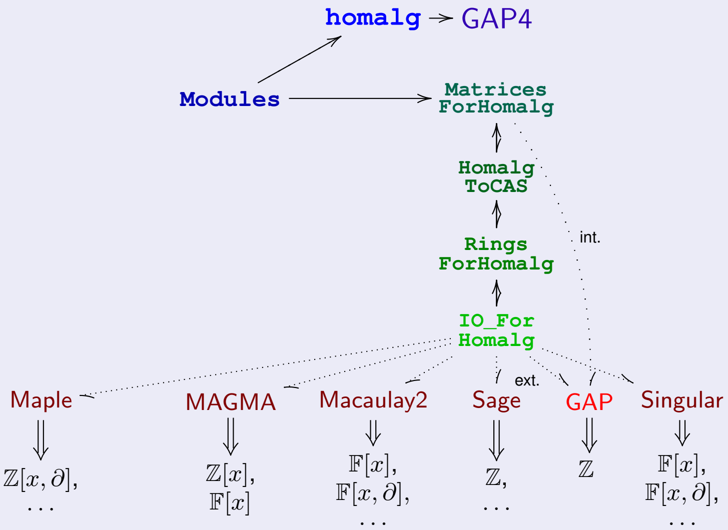
The homalg Project

IO_ForHomalg: Communicate via streams



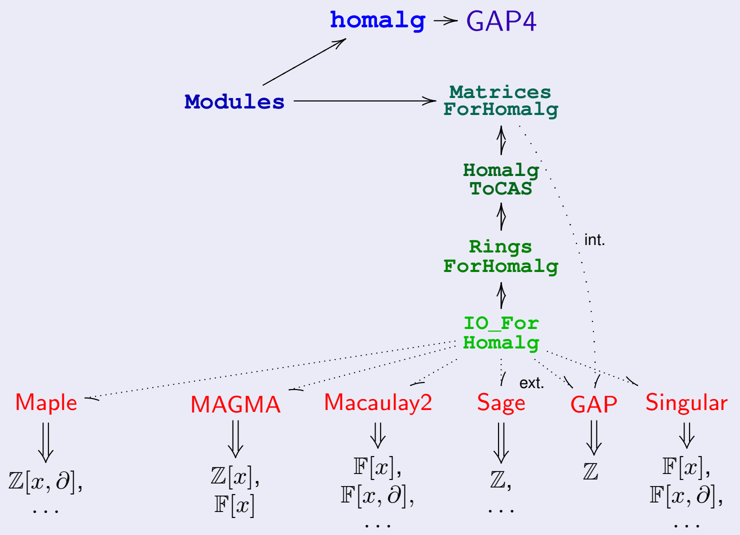
The homalg Project

IO_ForHomalg: Communicate via streams with various CA systems through their command line interface

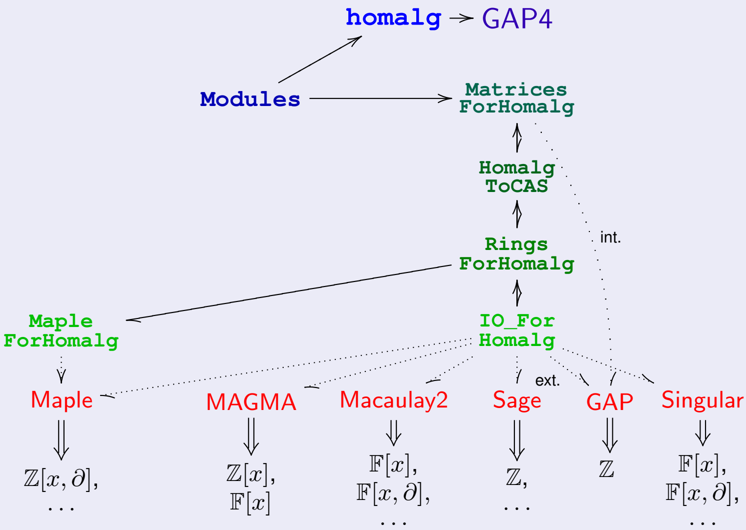


The homalg Project

External CASs host the matrices and GAP4 contains the higher logic → **Principle of least communication**

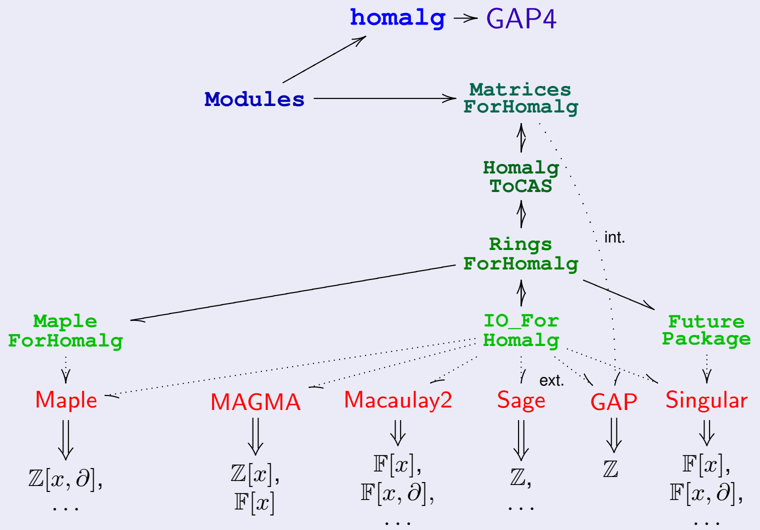


MapleForHomalg: Communicate with Maple's interpreter, shortcutting its command line interface

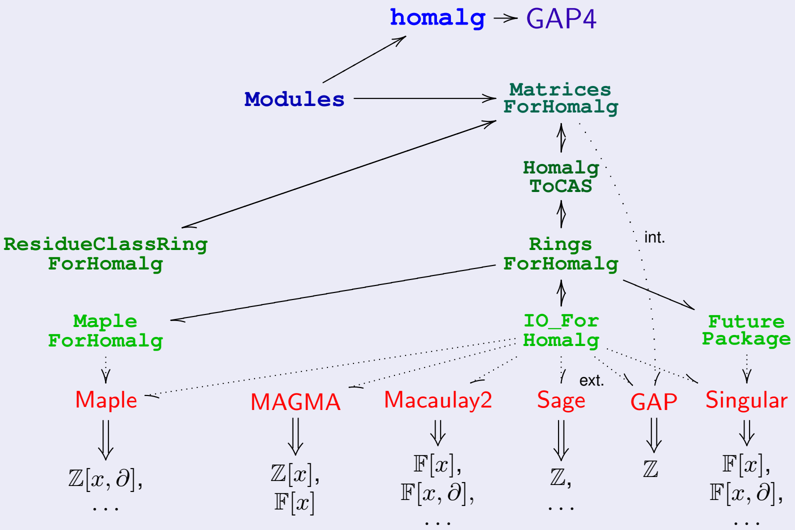


The homalg Project

Future: Communicate with interpreters of various CASs shortcutting their command line interface.



ResidueClassRingForHomalg: Support for residue class rings



Abstract Setting Applied: Localization

- R a commutative ring and $S \subseteq R$ multiplicatively closed.
  Localization: $S^{-1}R := \{\frac{r}{s} \mid r \in R, s \in S\}/_\sim$
- $S^{-1} = S^{-1}R \otimes_R$ is a functor from R-modules to $S^{-1}R$-modules.
- We treat the case: $\mathfrak{m} \in \mathrm{MaxSpec}(R)$, $S = R \setminus \mathfrak{m}$, $R_\mathfrak{m} := S^{-1}R$.
- Example: $R = K[x]$ and $\mathfrak{m} = \langle x \rangle$.
  Then $x \in \langle x - x^2 \rangle_{R_\mathfrak{m}}$, because $x = \frac{x - x^2}{1 - x}$.

Abstract Setting Applied: Localized Computable Rings

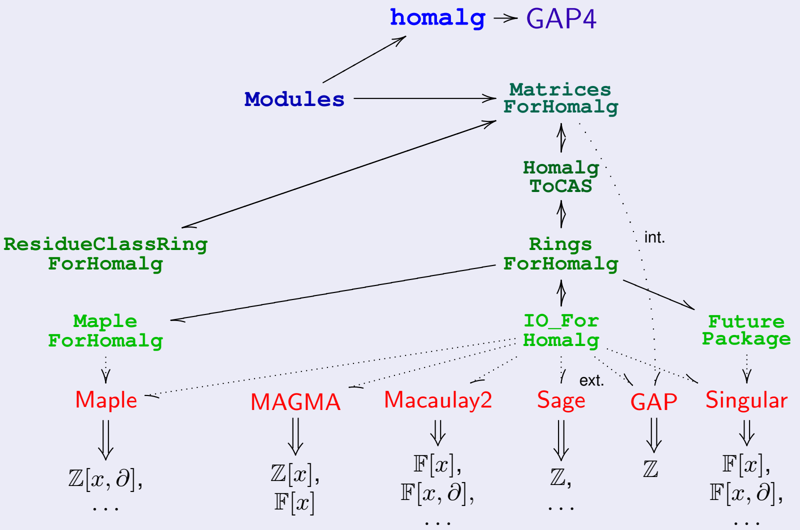
[BLH, Theorem 4.1]

Let R be a computable, commutative ring and $\mathfrak{m}$ a finitely generated maximal ideal in R. Then $R_\mathfrak{m} := (R \setminus \mathfrak{m})^{-1} R$ is computable.

- "Proof": $\text{Syzygies}_{R_\mathfrak{m}}(\,\cdot\,) = R_\mathfrak{m} \otimes_R \text{Syzygies}_R(\,\cdot\,)$
  The SubmoduleMembershipProblem of $R_\mathfrak{m}$ reduces to the SubmoduleMembershipProblem of R.

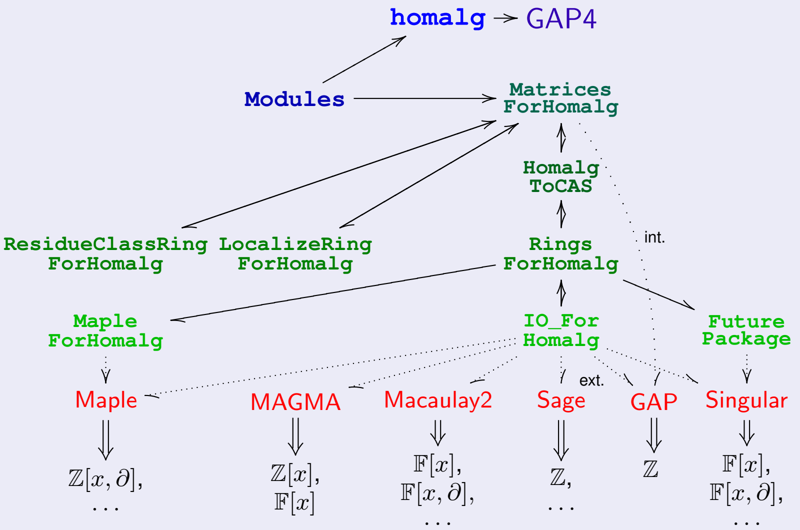
$(\square)$

ResidueClassRingForHomalg: Support for residue class rings



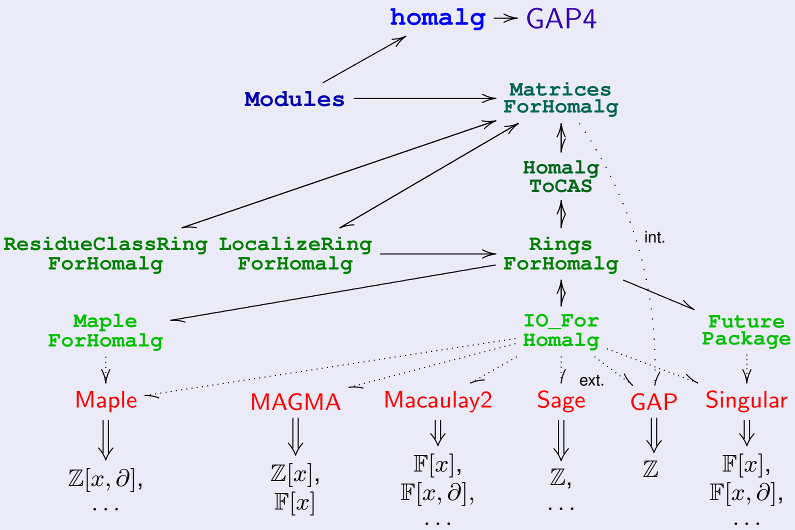
The homalg Project

LocalizeRingForHomalg: Localizations of commutative rings in homalg at maximal ideals.



The homalg Project

LocalizeRingForHomalg: Use MORA's algorithm in Singular to localize polynomial rings at maximal ideals.



Advantages of a well-structured Project

Easily extend `homalg` without bugging Mohamed

Just implemented a constructor for a ring in a new package without touching `homalg`'s code.

Advantages of a well-structured Project

Easily extend `homalg` without bugging Mohamed

Just implemented a constructor for a ring in a new package without touching `homalg`'s code.

- GAP manages numerator-denominator pairs.

Advantages of a well-structured Project

Easily extend `homalg` without bugging Mohamed

Just implemented a constructor for a ring in a new package without touching `homalg`'s code.

- GAP manages numerator-denominator pairs.
- The external system stores the numerator and denominator.

Advantages of a well-structured Project

Easily extend `homalg` without bugging Mohamed

Just implemented a constructor for a ring in a new package without touching `homalg`'s code.

- GAP manages numerator-denominator pairs.
- The external system stores the numerator and denominator.
- Before several operations we need a common denominator.
  Problem: This *greatly* enlarges entries of matrices.

Advantages of a well-structured Project

Easily extend `homalg` without bugging Mohamed

Just implemented a constructor for a ring in a new package without touching `homalg`'s code.

- GAP manages numerator-denominator pairs.
- The external system stores the numerator and denominator.
- Before several operations we need a common denominator.
  Problem: This *greatly* enlarges entries of matrices.

Advantages of `LocalizeRingForHomalg`

- Heuristics to suppress unnecessary denominators.

Advantages of a well-structured Project

Easily extend `homalg` without bugging Mohamed

Just implemented a constructor for a ring in a new package without touching `homalg`'s code.

- GAP manages numerator-denominator pairs.
- The external system stores the numerator and denominator.
- Before several operations we need a common denominator.
  Problem: This *greatly* enlarges entries of matrices.

Advantages of `LocalizeRingForHomalg`

- Heuristics to suppress unnecessary denominators.
- No (expensive) local computations in $R_{\mathfrak{m}}$.
  Instead compute in the "global" ring R.
  (Usually there are highly optimized algorithms for R.)

Advantages of a well-structured Project

Easily extend `homalg` without bugging Mohamed

Just implemented a constructor for a ring in a new package without touching `homalg`'s code.

- GAP manages numerator-denominator pairs.
- The external system stores the numerator and denominator.
- Before several operations we need a common denominator.
  Problem: This *greatly* enlarges entries of matrices.

Advantages of `LocalizeRingForHomalg`

- Heuristics to suppress unnecessary denominators.
- No (expensive) local computations in $R_{\mathfrak{m}}$.
  Instead compute in the "global" ring R.
  (Usually there are highly optimized algorithms for R.)
- Universal: Compatible with *any* commutative computable ring R.

Comparison

Singular outperforms Singular

Basing our approach to localization on $\mathbb{F}[x_1, \dots, x_n]$ outperforms MORA's algorithm for $\mathbb{F}[x_1, \dots, x_n]_{\langle x_1, \dots, x_n \rangle}$.

(Singular as computational back-end for both cases)

Example: SERRE's Intersection Formula

SERRE's formula of intersection multiplicity for two ideals $I, J \lhd R$ at a prime ideal $\mathfrak{p} \lhd R$:

$$i(I, J; \mathfrak{p}) = \sum_j (-1)^j \operatorname{length}\left(\operatorname{Tor}_j^{R_\mathfrak{p}}(R_\mathfrak{p}/I_\mathfrak{p}, R_\mathfrak{p}/J_\mathfrak{p}) \right)$$

Let $R := \mathbb{F}_5[x, y, z, v, w]$, $\mathfrak{p} = \mathfrak{m} = \langle x, y, z, v, w \rangle \lhd R$ maximal ideal and $R_0 = S_0 := R_\mathfrak{m}$.

Example: SERRE's Intersection Formula

SERRE's formula of intersection multiplicity for two ideals $I, J \lhd R$ at a prime ideal $\mathfrak{p} \lhd R$:

$$i(I,J;\mathfrak{p}) = \sum_j (-1)^j \underbrace{\text{length}}_{\text{MORA}} \left(\underbrace{\operatorname{Tor}_j^{R_\mathfrak{p}}(R_\mathfrak{p}/I_\mathfrak{p}, R_\mathfrak{p}/J_\mathfrak{p})}_{\text{MORA}} \right)$$

Let $R := \mathbb{F}_5[x,y,z,v,w]$, $\mathfrak{p} = \mathfrak{m} = \langle x,y,z,v,w \rangle \lhd R$ maximal ideal and $R_0 = S_0 := R_\mathfrak{m}$.

Example: SERRE's Intersection Formula

SERRE's formula of intersection multiplicity for two ideals $I, J \lhd R$ at a prime ideal $\mathfrak{p} \lhd R$:

$$i(I,J;\mathfrak{p}) = \sum_j (-1)^j \underbrace{\mathrm{length}}_{\text{MORA}} \left(\underbrace{\mathrm{Tor}_j^{R_\mathfrak{p}}(R_\mathfrak{p}/I_\mathfrak{p}, R_\mathfrak{p}/J_\mathfrak{p})}_{\cancel{\text{MORA}}} \right)$$

Let $R := \mathbb{F}_5[x,y,z,v,w]$, $\mathfrak{p} = \mathfrak{m} = \langle x,y,z,v,w \rangle \lhd R$ maximal ideal and $R_0 = S_0 := R_\mathfrak{m}$.

Example: SERRE's Intersection Formula

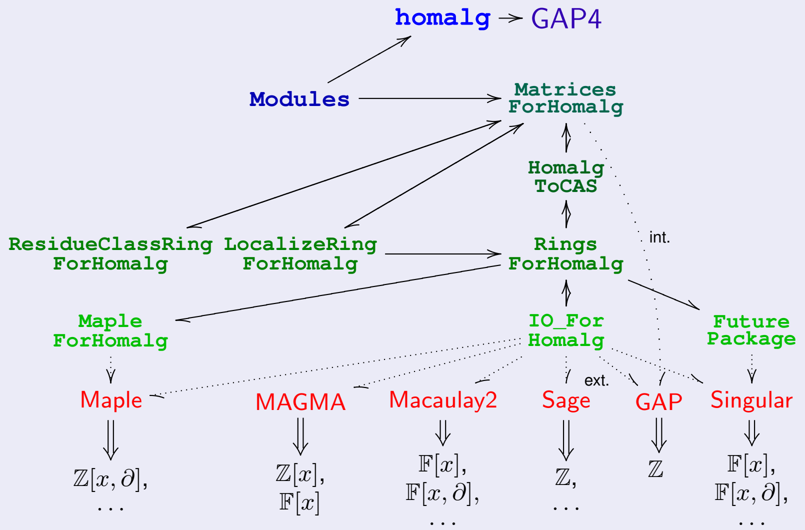
SERRE's formula of intersection multiplicity for two ideals $I, J \lhd R$ at a prime ideal $\mathfrak{p} \lhd R$:

$$i(I, J; \mathfrak{p}) = \sum_j (-1)^j \underbrace{\text{length}}_{\text{MORA}} \left(\underbrace{\text{Tor}_j^{R_\mathfrak{p}}(R_\mathfrak{p}/I_\mathfrak{p}, R_\mathfrak{p}/J_\mathfrak{p})}_{\text{BUCHBERGER}} \right)$$

Let $R := \mathbb{F}_5[x, y, z, v, w]$, $\mathfrak{p} = \mathfrak{m} = \langle x, y, z, v, w \rangle \lhd R$ maximal ideal and $R_0 = S_0 := R_\mathfrak{m}$.

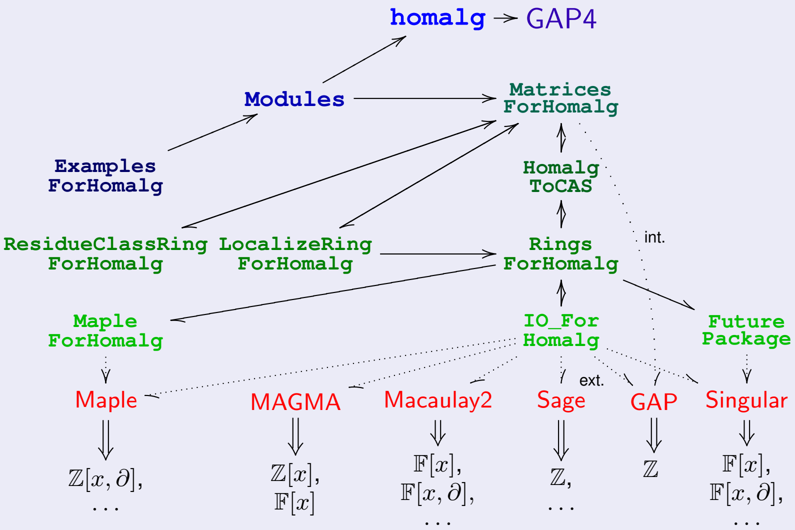
The `homalg` Project

`LocalizeRingForHomalg`: Use MORA's algorithm in Singular to localize polynomial rings at maximal ideals.



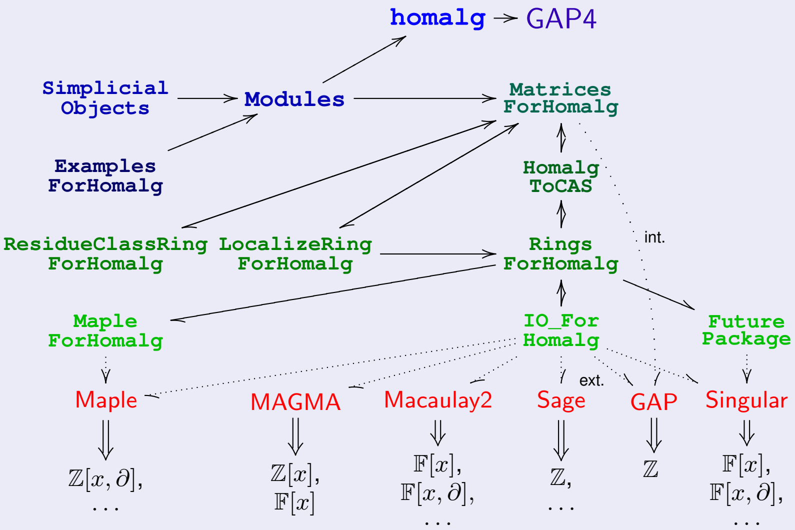
The homalg Project

ExamplesForHomalg: Computing-engine-independent example using homalg.



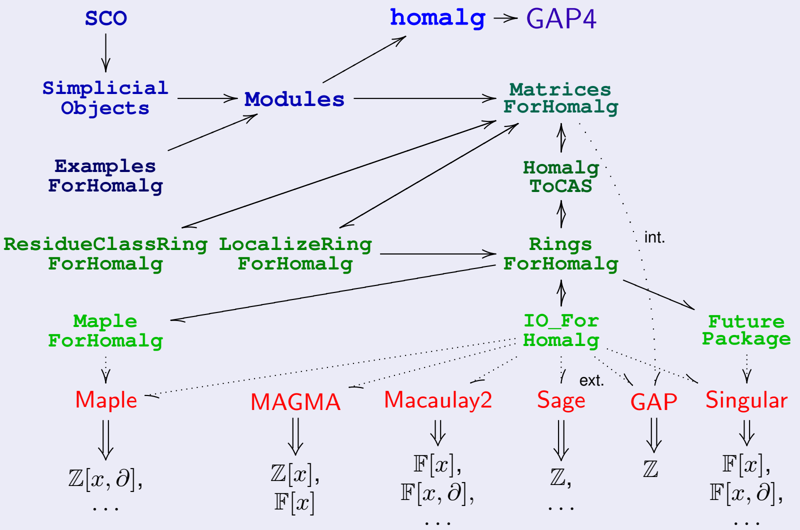
The homalg Project

SimplicialObjects: Simplicial objects for the homalg project

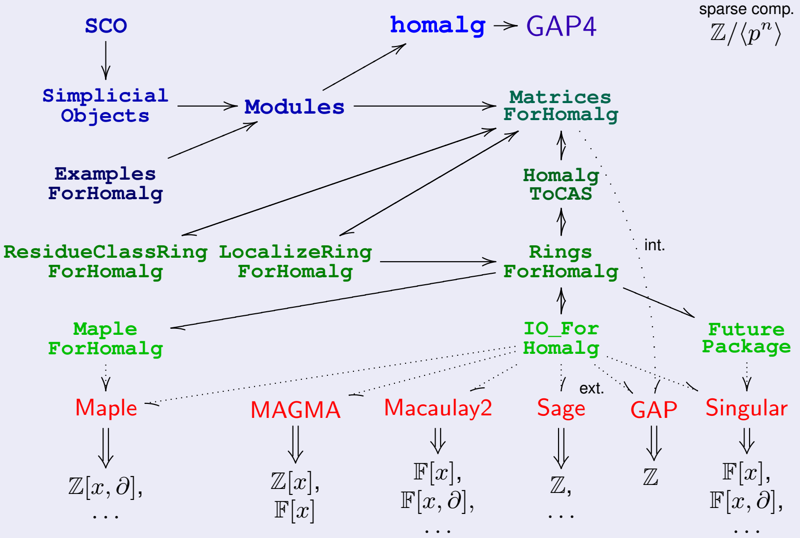


The homalg Project

SCO: Simplicial cohomology of orbifolds

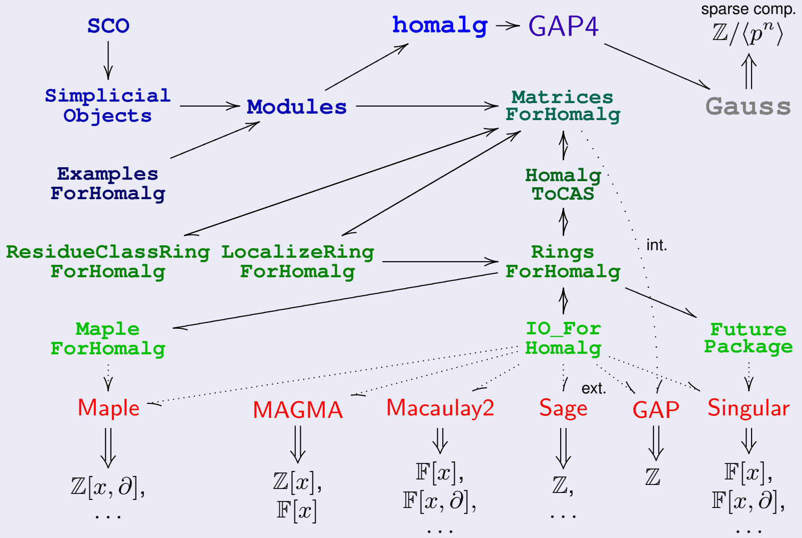


Sparse computations over p-adic numbers (a necessity not only for SCO)



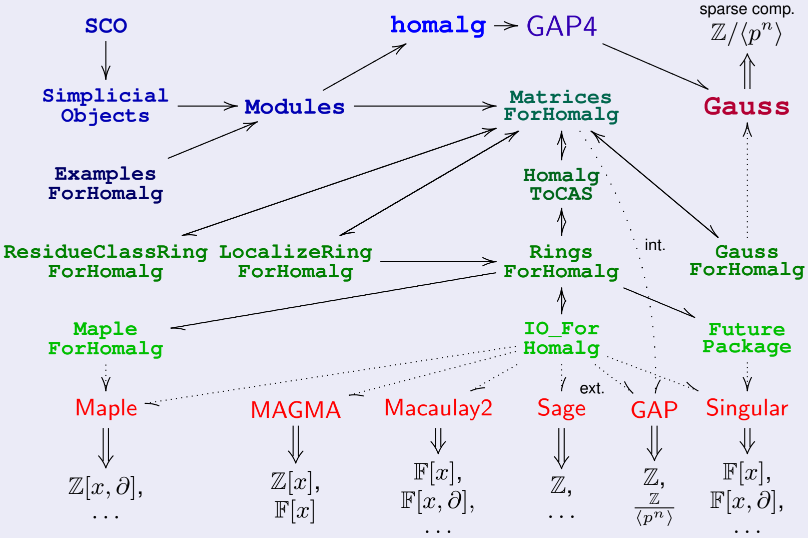
The homalg Project

Gauss: Added missing RREF to GAP4 for **sparse** matrices over $\mathbb{Z}/p^n\mathbb{Z}$ and $\mathbb{Q}$



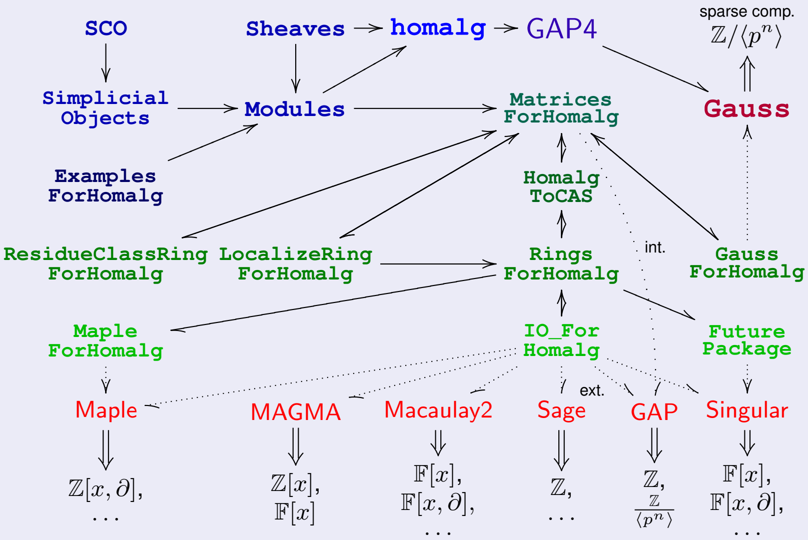
The homalg Project

GaussForHomalg: Linking Gauss and homalg



The homalg Project

Sheaves: Coherent sheaves of modules (& future projects: Advanced applications building upon homalg)



References

Mohamed Barakat and Markus Lange-Hegermann, *An Axiomatic Setup for Algorithmic Homological Algebra and an Alternative Approach to Localization*, to appear in Journal of Algebra and its Applications (`arXiv:1003.1943`).

The homalg project authors, *The* `homalg` *project*, 2003-2010, (`http://homalg.math.rwth-aachen.de/`).