Experiences with distributed polynomial computations

H. Schönemann

Overview

Connected Systems:

- Singular $\leftrightarrow$ Singular, using MP/MPP
- Mathematica $\leftrightarrow$ Singular, using MP/MPP + MathLink
- Singular $\leftrightarrow$ Maple,
  using ascii representation in the native Singular or Maple format

Purpose:

- storing data
- use other systems for computations - extended the functionality
- use other systems for computations - prototyping
- distributed computing

Different Needs:

- efficient : distributed computing
  $\rightarrow$ allow different representations of data
  (negotiable at runtime ?)
- general, extendable (at runtime) : prototyping
  $\rightarrow$ allow extending the context/new contexts. (corresponding to loading of additional libraries/procedures in CAS)
- allow partial understanding : typesetting, partial computations
- distinguish between structuring operators and functions:
  an OM tree should be completely parsable with only the basic contexts (for typesetting, user interface)
- allow storing of additional properties : retrieving stored data, checking results of partial computations

Proposals

- Inheritance of functions in different contexts
  Example: a type setting programm which does not understand the "Integer Context" (but the "Basis Arithmetic Context") should be able to to typeset "1+2" with "+" as generic-add.
  It should be clear how to find "basic/add-generic" from "integer/add-integer".

- Additional Objects in contexts
  (beside types, errors, constants, function/commands):
  properties (annotations), description of types.

Inheritance

- important for interfaces "non-mathematical" clients:
  (type setting, user interface): from which context is "+ in "1+2" ?
- allows partial implementations of contexts
  Example: a CAS may not be able to evalute $1 - sin^2(x)$ numerically, but to simplify it to $cos^2(x)$.
  Sometimes it is better to have this partial result than an error.

Properties/Annotations

- allows storing of additional informations (monomial order of polynomials,...)
- allows partial implementations of contexts
  Example: a CAS may not be able to evalute $factor(x^3 - y^3)\ mod\ 3$, but $factor(x^3 - y^3) = (x - y)(x^2 + xy + y^3)$.
  Sometimes it is better to have this partial result than an error - check the properties of the result.
- type information of types from (user defined) contexts
- distinction between "required" and "not required" annotations