PARI GP and sage

Jeroen Demeyer

Universiteit Gent

What is Sage?

- A free open source mathematics program with the goal: *Creating a viable free open source alternative to Magma, Maple, Mathematica and Matlab.*
- Much larger scope (all of mathematics) and larger project than PARI/GP.
- Started by William Stein (UWashington) for research on modular forms, first version in 2005.
- Programmed in Python (version 2.7.8) + Cython.
- Contains lots of existing mathematical programs and libraries: ATLAS, FLINT, GAP, Maxima, MPC, MPIR (fork of GMP), MPFR, NTL, Numpy, PARI/GP, PPL, R, Singular, . . .
  - Many of these are patched versions.

What is Cython?

- Cython (a fork of Pyrex) is a Python $\to$ C compiler, similar to what GP2C does for GP code.
- Goals:
  - Interface with C libraries, such as PARI. This was the initial goal of Pyrex.
  - Speed up Python code using C type declarations.
  - Support most of Python and C.
- Gives the best of both worlds: the speed of C with the flexibility of Python.

Use of PARI in Sage

- Sage contains PARI version 2.7.1 (+ patches).
- PARI is used for elementary number theory, factoring, large finite fields, some linear algebra, number fields, elliptic curves, some transcendental functions.
- Sage also contains:
  - C libraries using PARI: Cremona's eclib (elliptic curves), Rubinstein's lcalc (L-functions).
  - GP scripts by Buzzard (Hecke operators), Dokchitser (L-functions), Simon (2-descent).

The Sage $\leftrightarrow$ PARI interface

- Using Cython, Sage has an interface to PARI.
  - `GEN`s are wrapped in Python objects.
  - After each PARI call, the result is copied from the PARI stack.
  - For every PARI function, we need a Cython wrapper.
  - Usually, PARI functions are implemented as methods with the GP name, e.g. `pari(x).sin()` for the sine function.
- There is also an independent text-based interface to GP. This is mainly used for the GP scripts.
- Most of the interface was written for earlier versions of PARI (e.g. only a few months ago we changed Sage to use `t_FFELT` finite field elements everywhere instead of `t_POLMOD`).

Feature requests for PARI

- Port Simon's 2-descent scripts to PARI (and improve them).
- Improve `nfsplitting()`: reducible polynomials, early abort if degree is too large.
- Improve `polgalois()`: higher degrees (at least if group order is small), relative (over number fields).
- Better linear algebra (in particular over finite fields).

Other feature requests for PARI

- More releases (the difference between PARI 2.7.2 and master is huge).
- Less random results (e.g. the generator given by `bnfisprincipal()` changes all the time).
- Ensure PARI remains fast even if the stack is small.
- Support multiple libraries in the same executable all using PARI. (currently this works in Sage but it's not officially supported).
- Factoring with a callback function, called for every factor.
- Portable `writebin()` (write and read on different systems).
- Remove the `sizeof(long) == sizeof(void*)` limitation :)

Improving the interface

- Use `cb_pari_err_handle()` for much cleaner error handling (done, needs review at Sage #14894).
- Upgrade PARI and use new PARI features:
  - Use `parisizemax` instead of our own stack handler.
  - Remove limitations involving variable names: `varhigher()` and `varlower()`.
  - Lots of new mathematics. . .
- The file `gen.pyx` containing the Python interface to the PARI functions could be mostly auto-generated like in GP using `pari.desc` (but what about the documentation?).
- Avoid copying everything from the PARI stack (is there a real gain in actual use cases?).