t_ERROR
Errors as first class object

B. Allombert

IMB
CNRS/Université Bordeaux 1

25/01/2012

An example : Lenstra ECM

Let’s try to implement Lenstra ECM method in GP :

```
ecm(N, B = 1000!, nb = 100)=
{
  for(a = 1, nb,
    ellpow(ellinit([a,1]*Mod(1,N)),
           [0,1]*Mod(1,N), B))
}
```

We try to factor $2^{64} + 1$.

```
? ecm(2^64+1)
  ***   at top-level: ecm(2^64+1)
  ***                 ^-----------
  *** ellpow: impossible inverse modulo:
                 Mod(274177, 18446744073709551617).
  ***   Break loop: type 'break' to go back to GP
```

So we find the factor 274177 which is fine, but ending by an error is not very user friendly :

```
? for(i=60,70,print(ecm(2^i+1)))
  *** ellpow: impossible inverse modulo:
    Mod(129627369222792508, 1152921504606846977).
  ***   Break loop: type 'break' to go back to GP
break>
```

So we would like to trap the error but still get the error message somehow.

iferrname

So we introduce a new function `iferrname` :

```
ecm(N, B = 1000!, nb = 100)=
{
  for(a = 1, nb,
    iferrname("e_INTMOD",
      ellpow(ellinit([a,1]*Mod(1,N)),
                     [0,1]*Mod(1,N), B),
      E, return(gcd(lift(component(E,1)),N))))
}
? ecm(2^64+1)
%2 = 274177
```

How does that works ?

- The command `iferrname("e_INTMOD",expr1,E,expr2)` traps errors of type `"e_INTMOD"` in `expr1`. If an error occurs, `E` is set to the error data and `expr2` is evaluated. In that case, the error data is `Mod(274177, 18446744073709551617)`.
- The command `iferr(expr1,E,expr2)` is similar but catches every types of error, so you should better check you got one type error you expected.

What is E

The value E is actually a new kind of GEN object, of type `t_ERROR`.

- It has a name that can be read by `errname(E)`.
- It has a number of components depending on the error.
- It can be rethrown using `error(E)`.
- It is a normal object, that you can handle normally.

(Not so) silly example

```
? err=iferr(1/0,E,E);
? print(err)
error("division by a non-invertible object")
? error(err)
  *** error: division by a non-invertible object
  ***   Break loop: type 'break' to go back to GP
```

User errors

You can use `error()` to create user error :

```
? err=iferr(error("not a Mersenne number: ",17),
             E,E);
? component(err,1)
%2 = ["not a Mersenne number: ", 17]
? error(err)
  ***   user error: not a Mersenne number: 17
```

The new pari_err interface

In PARI 2.6 we renamed the error names like typeer to names like e_TYPE. For most names, we provide an helper function to avoid generating errors with incorrect data (e.g pari_err_TYPE).

The C interface to `t_ERROR`

- In the CATCH/TRY interface, the variable `global_err_data` contains the `t_ERROR` (or `NULL` if the error is a stack overflow).
- The global variable `iferr_env` allow to catch errors.
- `pari_err2str` allow to convert a `t_ERROR` to a `char *`.

Exemple of use of CATCH/TRY

```
pari_sp av = avma;
CATCH(e_INTMOD) {
  GEN x = gel(global_err_data,2);
  return gerepilecopy(av, x);
} TRY {
  powell(ell, P, B);
} ENDCATCH;
```

Note : `CATCH(CATCH_ALL)` allow to catch any errors.