CoCoALib - Slug #1042

LF curiously slow (breaking a poly into homog pieces)

10 Apr 2017 11:25 - John Abbott

Status:	Closed	**Start date:**	10 Apr 2017
Priority:	Low	**Due date:**	
Assignee:	John Abbott	**% Done:**	100%
Category:	Improving	**Estimated time:**	2.99 hours
Target version:	CoCoALib-0.99560	**Spent time:**	2.95 hours

Description

The following loop is curiously slow:

```
    while (!IsZero(fpow2))
      fpow2 -= LF(fpow2);
```

Anna has already made an improvement, but it ought to be faster.

Related issues:

Related to CoCoALib - Feature #1022: New "LF" function which is based on StdDeg	**New**	**06 Mar 2017**
Related to CoCoALib - Feature #1033: Split poly into homog parts	**Closed**	**17 Mar 2017**

History

#1 - 10 Apr 2017 11:29 - John Abbott

Here is a complete example:

```
  RingElem CutLF(RingElem& f)
  {
    const SparsePolyRing& P = owner(f);
    if (IsZero(f)) return f;
    RingElem ans(P);
    do
    {
      P->myMoveLMToBack(raw(ans), raw(f));
    }
    while (!IsZero(f) && (CmpWDeg(LPP(f), LPP(ans)) == 0));
    return ans;
  }

  void program()
  {
    GlobalManager CoCoAFoundations;

    ring P = NewPolyRing(RingQQ(), symbols("x,y,z"));
    RingElem f = ReadExpr(P,"x+y+z+1");
    RingElem fpow = power(f,199);
    RingElem fpow2 = fpow;
    const long n = NumTerms(fpow);
    long count = 0;
    // LOOP 1:
    double t0 = CpuTime();
    while (!IsZero(fpow))
    {
      RingElem lffpow = CutLF(fpow);
      count += NumTerms(lffpow);
    }
    cout << "loop1 time: " << CpuTime() - t0 << endl;
    cout << count -  n << endl;

    // LOOP 2:
    double t1 = CpuTime();
    while (!IsZero(fpow2))
      fpow2 -= LF(fpow2);
    cout << "loop2 time: " << CpuTime() - t1 << endl;
  }
```

LOOP 1 takes about 0.5s
LOOP 2 takes about 25s
While LOOP 2 will be slower because it is allocating memory, a factor of about 50 seems excessive.

#2 - 10 Apr 2017 11:29 - John Abbott

- Related to Feature #1022: New "LF" function which is based on StdDeg added

#3 - 10 Apr 2017 11:43 - John Abbott

I think this issue is relatively unimportant, hence the "low" priority.
I have put it on redmine just so that we do not forget it.

#4 - 03 Jul 2017 22:15 - John Abbott

- Status changed from New to Resolved

- Assignee set to John Abbott

- % Done changed from 0 to 80

I have checked in my implementation (almost the same as the one above, plus some arg checking).

QUESTION what should CutLF do if passed a zero poly as arg? Note that LF gives error.

#5 - 03 Jul 2017 22:21 - John Abbott

There was also a question about the name: I have called it CutLF. Another possibility could be MoveLF similar to MoveLM? This would require 2 args, one being the destination (and what should happen if the destination is not zero or in the wrong ring?)

#6 - 03 Jul 2017 22:22 - John Abbott

- Target version changed from CoCoALib-1.0 to CoCoALib-0.99560

#7 - 04 Jul 2017 15:20 - John Abbott

After some reflection and after chatting to Anna we have decided that CutLF should also give an error (like LF) when the arg is zero. I will check in shortly.

#8 - 06 Nov 2017 15:15 - John Abbott

- Status changed from Resolved to Closed

- % Done changed from 80 to 100

#9 - 06 Nov 2017 15:17 - John Abbott

- Estimated time set to 2.99 h

#10 - 08 Nov 2017 18:37 - John Abbott

- Related to Feature #1033: Split poly into homog parts added