CoCoALib - Slug #1105

Primary Decompositon (zero-dim) slow cases

02 Oct 2017 13:38 - John Abbott

Status:	Closed	**Start date:**	02 Oct 2017
Priority:	Normal	**Due date:**	
Assignee:	Anna Maria Bigatti	**% Done:**	100%
Category:	Improving	**Estimated time:**	0.00 hour
Target version:	CoCoALib-0.99650 November 2019	**Spent time:**	2.18 hours

Description

Here are some cases where PrimaryDecomposition0 in CoCoA-5.2.2 is slow:

```
time 6.9682
I := ideal(7*x*y^4 +9*x*y^3*z +4*z^5,  -6*x^5 +y^5 +4*x*y*z^2,  4*x^2*y^2*z -3*y^3*z^2 -2*x*z^4);

time 6.8479
I := ideal(-4*x^5 -5*y^5 -2*y*z,  -6*x*y^4 -6*x*z^4 +7*x^2,  -9*x^3*z^2 -7*x*y*z^3 -9*z^4);
```

PS I have ignored examples needing less than 5s on my computer.

2019-10 much better timings now

Related issues:

Related to CoCoALib - Feature #1178: New function: myPrimaryDecomposition_0dim	**Closed**	**06 Apr 2018**
Related to CoCoALib - Feature #1212: New function: GBasisByHomog	**Closed**	**05 Aug 2018**
Related to CoCoA-5 - Slug #1114: Some other examples for 0-dim radical	**Closed**	**31 Oct 2017**
Related to CoCoA-5 - Bug #1230: PrimaryDecomposition with lex ordering	**Closed**	**19 Oct 2018**
Related to CoCoALib - Slug #1337: PrimaryDecomposition: a interesting/patholo...	**In Progress**	**15 Oct 2019**

History

#1 - 02 Oct 2017 13:39 - John Abbott

The examples were generated by random search printing out successively slower cases.

#2 - 02 Oct 2017 14:08 - John Abbott

- Description updated

#3 - 02 Oct 2017 14:56 - John Abbott

Here are some more examples: each gen here has 4 terms

```
time 7.7093
I := ideal(2*y^4*z -5*y^3*z^2 -5*x*z^4 +6*x*z^2,  -8*x^5 -9*y^5 +7*x^2*z^2 +5*x*z^3,  -2*x^4*y +7*x*y^4 -8*z^4
 -4);
time 9.1628
I := ideal(-7*x^5 -3*x^2*y^3 -4*x^2*z -3*x,  4*x^2*y*z^2 -8*z^5 -8*z^3,  9*x*y^4 +7*y^5 -8*x^2*y^2 -3*y^3);
```

#4 - 31 Oct 2017 15:36 - John Abbott

Here are some examples I found lying around in a file:

```
I := ideal(2*y^5 +y*z^3 +2*z^4,  x^2*y*z^2 +2*x*y^2*z^2 +2*x^2*y*z,  x^5 +x*y^4 +2*y^4*z);  --> SLOW

I := ideal(-335989*y^5 -778478*x^4 -189325*x^2*z^2,  -403966*x*y^3*z -835406*z^4 -888572*x^3,  3497*y^3*z^2 +951857*y^4 -99361*x^2*z); --> > 1600s

I := ideal(3*x^5 -7*x^3*y*z -3*y*z^4 -3*y^4 +9*y^2*z^2 -5*y,  7*x^2*y*z^2 -7*y^3*z^2 +9*y*z^4 -9*y*z^3 +7*y*z^2 -9*z^3,  6*y^4*z -5*z^5 -5*x^3*y -3*x*y^3 -4*z^4); --> > 1000s

I := ideal(7*x*y^3*z -6*y^4*z -8*x*y*z^2 -9*y^2*z^2 +6*y*z,  -7*x^2*y^3 +9*x^2*y*z^2 -2*z^5 +8*x^4 +x^3 -6*x*y*z,  6*x^5 -3*x^3*y*z +8*x^2*y -8*x^2*z +6*y*z^2 -5*y); --> > 2400s, Singular took < 600s
```

#5 - 06 Apr 2018 20:31 - Anna Maria Bigatti

- Related to Feature #1178: New function: myPrimaryDecomposition_0dim added

#6 - 03 Aug 2018 17:55 - John Abbott

I have just tried the 9.16s example from comment 3. It now **does not finish** in a reasonable time for me.

Using verbosity, I see that the problem appears to be the computation of GBasis in line 707 of SparsePolyOps-IdealZeroDim.C
which takes ages.

#7 - 05 Aug 2018 16:41 - Anna Maria Bigatti

- Related to Feature #1212: New function: GBasisByHomog added

#8 - 05 Aug 2018 18:26 - Anna Maria Bigatti

- Description updated

- Assignee set to Anna Maria Bigatti

- Target version changed from CoCoA-5.?.? to CoCoA-5.3.0

- % Done changed from 0 to 30

I have implemented in cocoalib the function GBasisViaHomog (homog-->compute-->dehomog), as it was in the old cocoa5 package.
Should be much better now.

#9 - 05 Aug 2018 18:30 - Anna Maria Bigatti

- Project changed from CoCoA-5 to CoCoALib

- Category changed from enhancing/improving to Improving

- Target version changed from CoCoA-5.3.0 to CoCoALib-0.99650 November 2019

Moved to cocoalib, because now implemented in C++.

#10 - 06 Aug 2018 15:52 - Anna Maria Bigatti

- Related to Slug #1114: Some other examples for 0-dim radical added

#11 - 06 Aug 2018 15:53 - Anna Maria Bigatti

- Description updated

- Status changed from New to In Progress

#12 - 06 Aug 2018 16:06 - John Abbott

Some things still to do (or consider doing):

1. Compare speed with Singular
2. using CpuTimeOut option, attempt to compute GBases for the elements of the PrimDec (simple ones should simplify, hard ones do not become "black holes")

#13 - 01 Oct 2019 11:55 - John Abbott

- % Done changed from 30 to 50

Here is a short file which checks the computation times of the examples given above:

```
use P ::= QQ[x,y,z];

ListOfIdeals := [
         ideal(2*y^4*z -5*y^3*z^2 -5*x*z^4 +6*x*z^2,  -8*x^5 -9*y^5 +7*x^2*z^2 +5*x*z^3,  -2*x^4*y +7*x*y^4 -8
*z^4 -4),  -- time 7.7
         ideal(-7*x^5 -3*x^2*y^3 -4*x^2*z -3*x,  4*x^2*y*z^2 -8*z^5 -8*z^3,  9*x*y^4 +7*y^5 -8*x^2*y^2 -3*y^3)
, -- time 9.2
         ideal(2*y^5 +y*z^3 +2*z^4,  x^2*y*z^2 +2*x*y^2*z^2 +2*x^2*y*z,  x^5 +x*y^4 +2*y^4*z),  --> SLOW
         ideal(-335989*y^5 -778478*x^4 -189325*x^2*z^2,  -403966*x*y^3*z -835406*z^4 -888572*x^3,  3497*y^3*z^
2 +951857*y^4 -99361*x^2*z), --> time > 1600s
         ideal(3*x^5 -7*x^3*y*z -3*y*z^4 -3*y^4 +9*y^2*z^2 -5*y,  7*x^2*y*z^2 -7*y^3*z^2 +9*y*z^4 -9*y*z^3 +7*
y*z^2 -9*z^3,  6*y^4*z -5*z^5 -5*x^3*y -3*x*y^3 -4*z^4), --> time > 1000s
ideal(7*x*y^3*z -6*y^4*z -8*x*y*z^2 -9*y^2*z^2 +6*y*z,  -7*x^2*y^3 +9*x^2*y*z^2 -2*z^5 +8*x^4 +x^3 -6*x*y*z,
6*x^5 -3*x^3*y*z +8*x^2*y -8*x^2*z +6*y*z^2 -5*y) --> time > 2400s, Singular took < 600s
         ];

foreach I in ListOfIdeals do
  t0 := CpuTime();
  PDI := PrimaryDecomposition(I);
  println TimeFrom(t0);
endforeach;
```

I have just checked (2019-10-01) on my computer, and the times printed out were:

```
1.186
0.297
0.189
18.816
1.021
2.767
```

Should we close this issue? How much of the tasks in comment 12 has been done?

#14 - 02 Oct 2019 16:10 - John Abbott

This seems to be related to issue #1230

I have just tried the examples from comment 13, but changing the current ring to **QQ[x,y,z],lex**.
The computation times are much longer... well, the first example was still computing after 300+ seconds
(compared to 1.2s with degrevlex). I didn't try the other examples.

So I presume the code does not map into a ring with degrevlex ordering, compute there, and then map the result back. :-(

#15 - 02 Oct 2019 16:12 - John Abbott

- Related to Bug #1230: PrimaryDecomposition with lex ordering added

#16 - 03 Oct 2019 17:41 - Anna Maria Bigatti

- Description updated

- Status changed from In Progress to Feedback

- % Done changed from 50 to 90

I agree: now the timings are no longer that horrible.
Indeed not bad ;-)

We can close this issue.
Have we compared with Singular?

#17 - 10 Oct 2019 19:20 - Anna Maria Bigatti

- Status changed from Feedback to Closed

- % Done changed from 90 to 100

#18 - 15 Oct 2019 11:41 - Anna Maria Bigatti

- Related to Slug #1337: PrimaryDecomposition: a interesting/pathological example added