CoCoA-5 - Bug #226

HilbertBasis segv

10 Sep 2012 10:46 - Anna Maria Bigatti

Status:	Closed	**Start date:**	10 Sep 2012
Priority:	Normal	**Due date:**	
Assignee:	Anna Maria Bigatti	**% Done:**	100%
Category:	Incomplete function	**Estimated time:**	6.00 hours
Target version:	CoCoA-5.0.9	**Spent time:**	5.75 hours

Description

```
N := Mat([[0,1], [2,3], [3,1]]);
For I := 1 To 1000 Do HilbertBasis(N); EndFor;
```

Answer is the empty set; (sooner or later) crashes.
[2013-01-22 JAA gets SEGV; seems that only 3 iters are normally needed]

Related issues:

Related to CoCoALib - Feature #298: Valgrind: keep CoCoALib at 0 memory leaks	**Closed**	**28 Jan 2013**
Related to CoCoA-5 - Bug #1190: HilbertBasisKer: SEGV (again)	**Closed**	**19 Jun 2018**

History

#1 - 23 Jan 2013 12:34 - Anna Maria Bigatti

- Category set to Incomplete function

#2 - 23 Jan 2013 14:50 - John Abbott

If you run the two offending lines in a fresh CoCoA-5 then C5 reports an error that CurrentRing has no value! This cannot be correct behaviour since neither input nor output need any ring.

2013-01-24: JAA applied a quick fix to toric.cpkg5; the source code is rather ugly but we will not fix it because we expect it to be rewritten in C++ (sooner or later).

#3 - 24 Jan 2013 16:01 - John Abbott

- Status changed from New to In Progress

- Target version set to CoCoA-5.0.9

- % Done changed from 0 to 10

Here is some "simpler" CoCoA-5 code which triggers the problem:

```
LwM := matrix([[0, 1, 0, 0],
               [2, 3, 0, 0],
               [3, 1, 0, 0],
               [1, 0, 1, 0],
               [0, 1, 0, 1]] );

HBR ::= ZZ/(2)[x[1..2], w[1..2], t];
I := MatSequentialToric(HBR, LwM);  // repeat this line about 40 times...
```

GDB reports that the problem is in assigning the result (presumably in deleting the old value of I prior to copying the new value). It feels horribly like a ref count bug.

2013-01-25 one I got this:

```
Assertion failed: (o->refCount>=0), function intrusive_ptr_release, file LineProviders.H, line 114.
```

#4 - 24 Jan 2013 16:28 - Anna Maria Bigatti

I further reduced the error:

```
/**/ M := matrix([[0, 1, 0], [0, 1, 0], [1, 0, 0]]);
/**/ MatSequentialToric(NewPolyRing(QQ,SymbolRange("x",1,NumRows(M))), M);
```

2013-01-25 I misread the example; M is *not* an identity matrix.
Error due to the fact that MatSequentialToric is not checking input:
Toric does and correctly complains about the 0 column

#5 - 25 Jan 2013 14:47 - Anna Maria Bigatti

- Status changed from In Progress to Resolved

This was (one of) the bug(s?)

```
    Msmall[i] = (int*)malloc(nrows*sizeof(int));
```

should have been

```
    Msmall[i] = (int*)malloc(ncols*sizeof(int));
```

The excuse is that I guess I had changed the code from transposing M and using transposed(M) (MatrixView)

Truly impressive how such a bug proved so elusive and survived a CoCoASchool tutorial on HilbertBasis!!!!!!

Now I check some more tests.

#6 - 25 Jan 2013 15:40 - Anna Maria Bigatti

Still some error (I guess of the same nature): this time triggered by

```
Use P ::= QQ[w,x,y,z,a,b,c];
I := Ideal(x*z-y^2, x*w-y*z);
M := matrix( /*QQ*/
        [[1, 1, -1, 0, 0, 0],
         [1, 0, 0, 1, 0, 0],
         [0, 1, 0, 0, 1, 0],
         [0, 0, 1, 0, 0, 1]]
        );
for i := 1 to 1000 Do
  Print i, "T";  A := Toric(I, [y]);
  Print "H ";  H := Toric(P, M);
EndFor;
```

#7 - 25 Jan 2013 17:40 - John Abbott

Definitely **valgrind** time!

I suppose the bug never revealed itself before because malloc may well reserve slightly more space than you actually ask for (*e.g.* it may take a block whose length is a multiple of 16 bytes). Anyway valgrind should detect in a comprehensible way all cases where you read or write outside an allocated block.

#8 - 11 Feb 2013 11:49 - Anna Maria Bigatti

- Status changed from Resolved to Closed

- % Done changed from 10 to 100

#9 - 02 Apr 2014 18:30 - Anna Maria Bigatti

- Estimated time set to 6.00 h

#10 - 21 Jun 2018 18:18 - Anna Maria Bigatti

- Related to Bug #1190: HilbertBasisKer: SEGV (again) added