

CoCoALib - Slug #967

Improve saturate

10 Nov 2016 09:20 - Anna Maria Bigatti

Status:	Resolved	Start date:	10 Nov 2016
Priority:	Normal	Due date:	
Assignee:	Anna Maria Bigatti	% Done:	70%
Category:	Improving	Estimated time:	5.00 hours
Target version:	CoCoALib-0.99880	Spent time:	4.85 hours
Description			
saturate is slow (naive algorithm: repeated colon). Improve it.			
Related issues:			
Related to CoCoA-5 - Slug #948: radical is slow (compared to singular) on the...		Closed	18 Oct 2016
Related to CoCoALib - Feature #1619: Make saturate available in CoCoALib		Closed	15 Oct 2021
Related to CoCoALib - Bug #1790: saturate with zero ideals		Closed	13 Mar 2024

History

#1 - 10 Nov 2016 09:21 - Anna Maria Bigatti

- Related to Slug #948: radical is slow (compared to singular) on these examples added

#2 - 10 Nov 2016 09:21 - Anna Maria Bigatti

- Related to Support #942: Which names to use? Intersection/saturation vs intersect/saturate added

#3 - 10 Nov 2016 11:59 - Anna Maria Bigatti

- Description updated

I implemented another naive algorithm (elim(h, I+(f*h-1))), but seems even worse.

#4 - 10 Nov 2016 12:00 - Anna Maria Bigatti

- % Done changed from 0 to 20

#5 - 10 Nov 2016 16:54 - Anna Maria Bigatti

- % Done changed from 20 to 30

I have modified ComputeSaturation using the factorization (repeats saturation on the factors).

I don't know if that's generally better or not, it also depends on the cost of the factorization.

I worked on this example coming from radical: here the polynomial is actually a product of linear forms and the computation goes from 1600 to <10 seconds

```
use QQ[c[0..19]];
L := [
c[1]*c[6]*c[13] +c[1]*c[8]*c[11] -c[1]*c[8]*c[13]*c[18] -c[1]*c[9]*c[13]^2 -c[1]*c[12] -c[1]*c[13]*c[19] +c[1]*c[14]*c[18] +c[1]*c[16] -c[1]*c[18]^2 +c[3]*c[6]*c[11] -c[3]*c[6]*c[13]*c[18] -c[3]*c[7]*c[13]^2 -c[3]*c[8]*c[11]*c[18] +c[3]*c[8]*c[13]^2*c[19] -c[3]*c[8]*c[13]*c[16] +c[3]*c[8]*c[13]*c[18]^2 -2*c[3]*c[9]*c[11]*c[13] +c[3]*c[9]*c[13]^2*c[14] +c[3]*c[9]*c[13]*c[18]^2*c[18] -c[3]*c[10] -c[3]*c[11]*c[19] +c[3]*c[12]*c[18] -c[3]*c[13]*c[17] +2*c[3]*c[13]*c[18]*c[19] +c[3]*c[14]*c[16] -c[3]*c[14]*c[18]^2 -2*c[3]*c[16]*c[18] +c[3]*c[18]^3,
c[1]*c[7]*c[13] -c[1]*c[8]*c[13]*c[19] +c[1]*c[9]*c[11] -c[1]*c[9]*c[13]*c[14] +c[1]*c[17] -c[1]*c[18]*c[19] +c[3]*c[5]*c[13] -c[3]*c[6]*c[13]*c[19] +c[3]*c[7]*c[11] -c[3]*c[7]*c[13]*c[14] -c[3]*c[8]*c[11]*c[19] +c[3]*c[8]*c[13]*c[14] -c[3]*c[8]*c[19] -c[3]*c[8]*c[13]*c[17] +c[3]*c[8]*c[13]*c[18]*c[19] -c[3]*c[9]*c[11]*c[14] -c[3]*c[9]*c[12] +c[3]*c[9]*c[13]*c[19] +c[3]*c[9]*c[13]*c[18]^2*c[19] -c[3]*c[10] +c[3]*c[11]*c[19]^2 +c[3]*c[15] -c[3]*c[16]*c[19] -c[3]*c[17]*c[18] +c[3]*c[18]^2*c[19],
c[1]*c[6]*c[11] -c[1]*c[8]*c[13]*c[16] -c[1]*c[9]*c[11]*c[13] -c[1]*c[10] -c[1]*c[11]*c[19] +c[1]*c[14]*c[16] -c[1]*c[16]*c[18] -c[3]*c[6]*c[13]*c[16] -c[3]*c[7]*c[11]*c[13] +c[3]*c[8]*c[11]*c[13]*c[19] -c[3]*c[8]*c[11] +c[3]*c[8]*c[13]*c[16] -c[3]*c[9]*c[11]*c[18] +c[3]*c[9]*c[11]*c[19] -c[3]*c[9]*c[13]*c[14] -c[3]*c[10] +c[3]*c[11]*c[19]^2 +c[3]*c[13]*c[14] +c[3]*c[13]*c[18]^2*c[16] -c[3]*c[16]*c[18] -c[3]*c[17]*c[19] +c[3]*c[18]^2*c[19]]
```

```

[3]*c[11]*c[17] +c[3]*c[11]*c[18]*c[19] +c[3]*c[12]*c[16] +c[3]*c[13]*c[16]*c[19] -c[3]*c[14]*c[16]*c[18] -c[3]*c[16]^2 +c[3]*c[16]*c[18]^2,
c[1]*c[5]*c[13] +c[1]*c[7]*c[11] -c[1]*c[8]*c[13]*c[17] -c[1]*c[9]*c[12]*c[13] -c[1]*c[12]*c[19] +c[1]*c[14]*c[17] +c[1]*c[15]*c[16]*c[18] -c[1]*c[18] -c[1]*c[19] -c[1]*c[20]*c[13] +c[3]*c[5]*c[11] -c[3]*c[6]*c[13]*c[17] -c[3]*c[7]*c[12]*c[13] -c[3]*c[8]*c[11]*c[17] +c[3]*c[8]*c[13]*c[19] -c[3]*c[8]*c[15] +c[3]*c[8]*c[17]*c[18] -c[3]*c[9]*c[10]*c[13] -c[3]*c[9]*c[11]*c[12] +c[3]*c[9]*c[13]*c[14] +c[3]*c[9]*c[17]*c[18] -c[3]*c[10]*c[13] -c[3]*c[11]*c[17] +c[3]*c[12]*c[13]*c[19] +c[3]*c[14]*c[15] -c[3]*c[14]*c[17]*c[19] +c[3]*c[15]*c[18] -c[3]*c[16]*c[17] +c[3]*c[17]*c[18]^2,
c[1]*c[5]*c[11] -c[1]*c[8]*c[13]*c[15] -c[1]*c[9]*c[10]*c[13] -c[1]*c[10]*c[19] +c[1]*c[14]*c[15] -c[1]*c[15]*c[18] -c[3]*c[6]*c[13]*c[15] -c[3]*c[7]*c[10]*c[13] +c[3]*c[8]*c[10]*c[13]*c[19] -c[3]*c[8]*c[11]*c[15] +c[3]*c[8]*c[13]*c[18] -c[3]*c[9]*c[10]*c[11] +c[3]*c[9]*c[10]*c[13]*c[14] +c[3]*c[9]*c[13]*c[15] -c[3]*c[10]*c[17] +c[3]*c[10]*c[18]*c[19] +c[3]*c[12]*c[15]*c[19] +c[3]*c[14]*c[15] -c[3]*c[14]*c[17]*c[19] +c[3]*c[15]*c[18] -c[3]*c[15]*c[16] +c[3]*c[17]*c[18]^2,
-c[6]*c[13] -c[8]*c[11] +c[8]*c[13]*c[18] +c[9]*c[13]^2 +c[12] +c[13]*c[19] -c[14]*c[18] -c[16] +c[18]^2,
-c[7]*c[13] +c[8]*c[13]*c[19] -c[9]*c[11] +c[9]*c[13]*c[14] -c[17] +c[18]*c[19],
-c[6]*c[11] +c[8]*c[13]*c[16] +c[9]*c[11]*c[13] +c[10] +c[11]*c[19] -c[14]*c[16] +c[16]*c[18],
-c[5]*c[13] -c[7]*c[11] +c[8]*c[13]*c[17] +c[9]*c[12]*c[13] +c[12]*c[19] -c[14]*c[17] -c[15] +c[17]*c[18],
-c[5]*c[11] +c[8]*c[13]*c[15] +c[9]*c[10]*c[13] +c[10]*c[19] -c[14]*c[15] +c[15]*c[18],
-c[1]*c[7]*c[13] +c[1]*c[8]*c[13]*c[19] -c[1]*c[9]*c[11] +c[1]*c[9]*c[13]*c[14] -c[1]*c[13]*c[17] +c[1]*c[18]*c[19]
-c[3]*c[5]*c[13] +c[3]*c[6]*c[13]*c[19] -c[3]*c[7]*c[11] +c[3]*c[7]*c[13]*c[14] +c[3]*c[8]*c[11]*c[19] -c[3]*c[8]*c[13]*c[14]*c[19] +c[3]*c[8]*c[13]*c[17] -c[3]*c[8]*c[18]*c[19] +c[3]*c[9]*c[11]*c[14] +c[3]*c[9]*c[12]*c[13] -c[3]*c[9]*c[13]*c[19] -c[3]*c[10]*c[17] -c[3]*c[10]*c[18]*c[19] +c[3]*c[12]*c[15]*c[19] +c[3]*c[14]*c[15] -c[3]*c[14]*c[17]*c[19] +c[3]*c[15]*c[18] -c[3]*c[15]*c[16] +c[3]*c[17]*c[18]^2,
c[1]*c[5] -c[1]*c[6]*c[19] -c[1]*c[7]*c[14] +c[1]*c[7]*c[18] +c[1]*c[8]*c[14]*c[19] -c[1]*c[8]*c[17] -c[1]*c[9]*c[12]*c[12] +c[1]*c[9]*c[19]^2 -c[1]*c[9]*c[14]^2 -c[1]*c[9]*c[18] +c[1]*c[9]*c[16] +c[1]*c[19]^2 -c[3]*c[5]*c[14] +c[3]*c[5]*c[18] +c[3]*c[6]*c[14]*c[19] -c[3]*c[6]*c[17] -c[3]*c[7]*c[12] +c[3]*c[7]*c[13]*c[19] +c[3]*c[7]*c[14]^2 -c[3]*c[7]*c[14]*c[18] +c[3]*c[7]*c[16] +c[3]*c[8]*c[12]*c[19] -c[3]*c[8]*c[13]*c[19]^2 -c[3]*c[8]*c[14]*c[19] +c[3]*c[8]*c[15] -c[3]*c[8]*c[9]*c[10] +c[3]*c[9]*c[11]*c[19] +2*c[3]*c[9]*c[12]*c[14] -c[3]*c[9]*c[12]*c[18] -c[3]*c[9]*c[12]*c[19] -2*c[3]*c[9]*c[13]*c[14]*c[19] +c[3]*c[9]*c[13]*c[17] -c[3]*c[9]*c[14]^3 +c[3]*c[9]*c[14]*c[18] -c[3]*c[9]*c[14]*c[16] -c[3]*c[14]*c[19]^2 +2*c[3]*c[17]*c[19] -c[3]*c[18]*c[19]^2,
-c[1]*c[7]*c[11] +c[1]*c[8]*c[11]*c[19] +c[1]*c[9]*c[11]*c[14] -c[1]*c[9]*c[11]*c[18] +c[1]*c[9]*c[13]*c[16] -c[1]*c[15]*c[15] +c[1]*c[16]*c[19] -c[3]*c[5]*c[11] +c[3]*c[6]*c[11]*c[19] +c[3]*c[7]*c[11]*c[14] -c[3]*c[7]*c[11]*c[18] +c[3]*c[7]*c[13]*c[16] -c[3]*c[8]*c[11]*c[19] +c[3]*c[8]*c[11]*c[17] -c[3]*c[8]*c[13]*c[16]*c[19] +c[3]*c[9]*c[11]*c[12] -c[3]*c[9]*c[11]*c[13]*c[19] -c[3]*c[9]*c[11]*c[14]^2 +c[3]*c[9]*c[11]*c[14]*c[18] -c[3]*c[9]*c[13]*c[14] -c[3]*c[11]*c[19]^2 +c[3]*c[12]*c[16]*c[19] -c[3]*c[12]*c[17]*c[19],
c[1]*c[5]*c[18] -c[1]*c[6]*c[17] -c[1]*c[7]*c[12] +c[1]*c[7]*c[16] +c[1]*c[8]*c[12]*c[19] -c[1]*c[8]*c[15] -c[1]*c[9]*c[10] +c[1]*c[9]*c[12]*c[14] -c[1]*c[9]*c[18] -c[1]*c[9]*c[12]*c[14] -c[1]*c[9]*c[18] +c[1]*c[9]*c[13]*c[17] +c[1]*c[17]*c[19] -c[3]*c[5]*c[12] +c[3]*c[5]*c[16] +c[3]*c[6]*c[12]*c[19] -c[3]*c[6]*c[15] -c[3]*c[7]*c[10] +c[3]*c[7]*c[12]*c[14] -c[3]*c[7]*c[12]*c[18] +c[3]*c[7]*c[13]*c[17] +c[3]*c[8]*c[10]*c[19] -c[3]*c[8]*c[12]*c[19] +c[3]*c[8]*c[14]*c[19] -c[3]*c[8]*c[14]*c[17] -c[3]*c[8]*c[18]*c[19] +c[3]*c[9]*c[10]*c[19] -c[3]*c[9]*c[12]*c[19]^2 -c[3]*c[9]*c[12]*c[13]*c[19] -c[3]*c[9]*c[12]*c[14]*c[19] +c[3]*c[9]*c[13]*c[17]*c[19] +c[3]*c[9]*c[14]*c[18] -c[3]*c[9]*c[15]*c[19] +c[3]*c[11]*c[17]*c[19],
c[1]*c[5]*c[16] -c[1]*c[6]*c[15] -c[1]*c[7]*c[10] +c[1]*c[8]*c[10]*c[19] +c[1]*c[9]*c[10]*c[14] -c[1]*c[9]*c[19]*c[1] +c[1]*c[10]*c[18] +c[1]*c[9]*c[13]*c[15] +c[1]*c[15]*c[19] -c[3]*c[5]*c[15] -c[3]*c[6]*c[10]*c[18] +c[3]*c[7]*c[13]*c[15] -c[3]*c[8]*c[8]*c[10]*c[19] +c[3]*c[8]*c[14]*c[19] +c[3]*c[8]*c[10]*c[17] -c[3]*c[8]*c[13]*c[17] +c[3]*c[9]*c[10]*c[19] -c[3]*c[9]*c[12]*c[19]^2 -c[3]*c[9]*c[12]*c[13]*c[19] -c[3]*c[9]*c[12]*c[14]*c[19] +c[3]*c[9]*c[13]*c[17]*c[19] +c[3]*c[9]*c[14]*c[18] -c[3]*c[9]*c[15]*c[19] +c[3]*c[11]*c[17]*c[19],
c[7]*c[13] -c[8]*c[13]*c[19] +c[9]*c[11] -c[9]*c[13]*c[14] +c[17] -c[18]*c[19],
-c[5]*c[6]*c[19] +c[7]*c[14] -c[7]*c[18] -c[8]*c[14]*c[19] +c[8]*c[17] +c[9]*c[12] -c[9]*c[13]*c[19] -c[9]*c[14]^2 +c[9]*c[14]*c[18] -c[9]*c[16] -c[19]^2,
c[7]*c[11] -c[8]*c[11]*c[19] -c[9]*c[11]*c[14] +c[9]*c[11]*c[18] -c[9]*c[13]*c[16] +c[15] -c[16]*c[19],
-c[5]*c[18] +c[6]*c[17] +c[7]*c[12] -c[7]*c[16] -c[8]*c[12]*c[19] +c[8]*c[15] +c[9]*c[10] -c[9]*c[12]*c[14] +c[9]*c[12]*c[18] -c[9]*c[13]*c[17] -c[17]*c[19],
-c[5]*c[16] +c[6]*c[15] +c[7]*c[10] -c[8]*c[10]*c[19] -c[9]*c[10]*c[14] +c[9]*c[10]*c[18] -c[9]*c[13]*c[15] -c[15]*c[19];

```

```

/**/ G := c[11]*c[14]*c[16]^2*c[17]^2*c[18]^2*c[19]^2 -c[11]*c[14]*c[15]*c[16]*c[17]*c[18]^3*c[19]^2 -c[11]*c[14]*c[16]^3*c[17]*c[18]*c[19]^3 -c[11]*c[14]*c[16]^2*c[17]^3*c[18]*c[19] +c[11]*c[14]*c[15]*c[16]*c[17]^2*c[18]^2*c[19]^2 -c[11]*c[14]*c[15]^2*c[16]*c[17]*c[18]^2*c[19];

```

```

I := ideal(L);
t := CpuTime(); S := saturate(I, ideal(G)); TimeFrom(t);

```

#6 - 10 Nov 2016 17:01 - Anna Maria Bigatti

I see that there is an involutive saturation.
How does that work in general? should we use that instead of GBases?

#7 - 02 Mar 2017 10:57 - John Abbott

- *Target version changed from CoCoALib-0.99550 spring 2017 to CoCoALib-0.99560*

#8 - 06 Nov 2017 14:09 - John Abbott

- *Target version changed from CoCoALib-0.99560 to CoCoALib-0.99600*

#9 - 02 Aug 2018 16:04 - Anna Maria Bigatti

- *Status changed from New to Resolved*
- *Target version changed from CoCoALib-0.99600 to CoCoALib-0.99650 November 2019*

The fix I did is not bad for that class of examples, and to say it is slow we should make proper comparisons....
postponing again.

#10 - 01 Oct 2019 11:40 - John Abbott

- *Target version changed from CoCoALib-0.99650 November 2019 to CoCoALib-0.99700*

#11 - 20 Jan 2020 09:27 - John Abbott

- *Target version changed from CoCoALib-0.99700 to CoCoALib-0.99800*

#12 - 04 Feb 2022 22:23 - John Abbott

- *Target version changed from CoCoALib-0.99800 to CoCoALib-0.99850*

#13 - 15 May 2023 20:32 - John Abbott

Let P be a poly ring with indets x[1]..x[n] and y[1]..y[m].
Suppose the gens of ideal I involve only x indets. If I want to saturate w.r.t. an irred poly which involves some y indets,
is there a short cut?
In particular, if I saturate w.r.t. y[1] then there is nothing to compute. Presumably the same if I saturate w.r.t. an ideal J which has gens involving only
y indets [!!'m guessing here!].
Are there theoretical results about this? If not, can we prove something.

Orig context: to homogenize an ideal, we can homogenize the gens then saturate w.r.t. the homogenizing indets. Now if the orig polys are already
homogeneous then we will be in the situation described above. Of course, we could also just do IsHomog on each gen... (or compute a RGB then
IsHomog on its elems).

#14 - 16 May 2023 09:24 - John Abbott

My suggestion above is not quite right.
The poly which I am saturating with respect to must be reduced modulo I.

I also note that CoCoALib has a function ContentFreeFactor (or similar name) which computes a factorization based on which indets appear; the
factorization is obtained by repeatedly computing the content (essentially GCD computations).

Anyway, the particular case I had in mind was that the poly "underneath" in the saturation is just a product of indets.
So any indets which do not appear in the gens of I can just be skipped.

#15 - 13 Mar 2024 19:23 - John Abbott

- Related to Feature #1619: Make saturate available in CoCoALib added

#16 - 13 Mar 2024 19:27 - John Abbott

- Related to Bug #1790: saturate with zero ideals added

#17 - 16 Mar 2024 21:24 - John Abbott

- % Done changed from 30 to 70

I have just tried the example from #note-5, and it took just less than 1s on my computer (with v0.99823). I have increased the %done to 70, since the status is "resolved".

#18 - 21 Mar 2024 17:57 - Anna Maria Bigatti

- Related to deleted (Support #942: Which names to use? Intersection/saturation vs intersect/saturate)

#19 - 21 Mar 2024 18:02 - Anna Maria Bigatti

- Target version changed from CoCoALib-0.99850 to CoCoALib-0.99880

I also added the factorization in the case of a monomial (similar test as in [#967-5](#), now in test0saturate.cocoa5).

Yet to do: saturation for homog ideal: I think I had some code by John I needed to integrate into cocoalib (find it! and its examples). That will take some work (and will be worth it!). Postponing to next version.

#20 - 22 Mar 2024 10:18 - Anna Maria Bigatti

Different situations to consider (from [#1619-8](#))

- monomial
- homog
- principal
- univariate
- general