CoCoALib - Design #1798

Computing in sub polyring

22 Mar 2024 09:23 - John Abbott

Status:	New	**Start date:**	22 Mar 2024
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Improving	**Estimated time:**	0.00 hour
Target version:	CoCoALib-0.99880	**Spent time:**	0.00 hour

Description

Investigate whether it is a good idea to adapt certain operations to compute in a sub polyring (without unnecessary indets)
Which functions would benefit?
How to specify the sub polyring and mapping back and forth.

Related issues:

Related to CoCoALib - Bug #1641: gcd does not recognize univariate input	**Closed**	**20 Dec 2021**
Related to CoCoALib - Slug #1057: Slug: Polynomial ring contructor slow with ...	**In Progress**	**04 May 2017**

History

#1 - 22 Mar 2024 09:25 - John Abbott

- Related to Bug #1641: gcd does not recognize univariate input added

#2 - 22 Mar 2024 09:26 - Anna Maria Bigatti

- Related to Slug #1057: Slug: Polynomial ring contructor slow with (big) matrix ordering added

#3 - 22 Mar 2024 09:27 - Anna Maria Bigatti

From #1641-11
It seems it's the "high number of variables" problem, and syz itself is quite fast: this examples takes ~4s on my computer.
Try with verbosity (first check out by verbosity additions):

```
/**/ SetVerbosityLevel(130);
/**/ /**/ use ZZ/(19)[u[1..400],x,y[1..400]];
/**/ f1 := x^4 -x^2 +7*x -7;
/**/ f2 := x^4 +4*x^3 -6*x^2 -6*x +7;
/**/ g1 := subst(f1,x,u[1]-y[1]);
/**/ g2 := subst(f2,x,u[1]-y[1]);
/**/ t0 := CpuTime(); syz([g1,g1]); TimeFrom(t0);
```

#4 - 22 Mar 2024 09:30 - Anna Maria Bigatti

The test in note-3 would be faster if called in the minimal sub-polyring.