CoCoALib - Feature #1381

Type ideal for ZZ[x[1..n]]

06 Jan 2020 06:46 - Elisa Palezzato

Status:	In Progress	**Start date:**	06 Jan 2020
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	10%
Category:	Data Structures	**Estimated time:**	0.00 hour
Target version:	CoCoALib-0.99880	**Spent time:**	2.45 hours

Description

Is it possible to add the type ideal for the ring ZZ[x[1..n]]?

History

#1 - 07 Jan 2020 14:28 - John Abbott

- Project changed from CoCoA to CoCoALib

- Category set to Data Structures

- Target version set to CoCoALib-1.0

JAA moved this to CoCoALib.

I think this will not be possible before the CoCoA School in early March (or, at least, we cannot guarantee anything before then).

Probably the type name should be something like IdealOverPID; maybe even just IdealOverZZ for a first version (but hoping that most of the impl is general enough to work for PIDs).

#2 - 07 Jan 2020 14:48 - Anna Maria Bigatti

I think the question was intended as: "can we let the user define an ideal over ZZ[x[1..n]]?"
Now it gives an error:

```
/**/ use ZZ[x,y];
/**/ I := ideal(2*x);
--> ERROR: ERR:NYI ideal of polynomials with coeffs not in a field
--> [CoCoALib] ideal(SparsePolyRing, gens)
--> I := ideal(2*x);
-->      ^^^^^^^^^^
```

In fact, even though we cannot (yet!) do much with them, there are possibile functions: sum, product, gens, ... so we should just move the blocking check into the "gbasis" function.

#3 - 21 Feb 2020 14:54 - John Abbott

- Target version changed from CoCoALib-1.0 to CoCoALib-0.99850

#4 - 27 Apr 2021 13:56 - John Abbott

- Status changed from New to In Progress

- % Done changed from 0 to 10

The relevant source code seems to be around line 364 of SparsePolyOps-ideal.C.

I'll try disabling the check inside the pseudo-ctor and see what happens... :-)

#5 - 27 Apr 2021 14:05 - John Abbott

The CoCoA-5 tests all passed, and I succeeded in creating an ideal in ZZ[x,y,z].

Next thing is to check that some trivial operations work:

- I+J, I*J
- r*I
- gens(I), NumGens(I)
- IsZero(I)
- other ops?

#6 - 27 Apr 2021 20:22 - John Abbott

A quick test of ideal product suggests that it does not work properly in ZZ[...]

```
/**/ use ZZ[x,y,z];
/**/ I := ideal(2*x, x^2, y^3);
/**/ I;
ideal(2*x,  x^2,  y^3)
/**/ J := ideal(4*x, x^3, 2*y,3*z);
/**/ I*J;
ideal(x*z,  x*y,  x^2,  y^3*z,  y^4)  // where have all the coeffs gone?
```

#7 - 27 Apr 2021 21:51 - John Abbott

I suspect the example in the previous note is wrong because an ideal is a recognized as "monomial over field" without actually checking that the coeff ring is a field.

#8 - 27 Apr 2021 22:45 - John Abbott

I think I have fixed ideal product now: it calls the "clever monomial short-cut" only if the coeff ring is a field (and the other criteria are met).
Source code: SparsePolyOps-ideal.C around line 575.

Output is now:

```
ideal(8*x^2,  2*x^4,  4*x*y,  6*x*z,  4*x^3,  x^5,  2*x^2*y,  3*x^2*z,  4*x*y^3,  x^3*y^3,  2*y^4,  3*y^3*z)
```

Could it be worth having special monomial ideals with coeffs in ZZ (or a PID)? Not sure.

#9 - 28 Apr 2021 10:57 - John Abbott

Here is my first *ad hoc* test:

```
use ZZ[x,y,z];
I := ideal(2*x, x^2, y^3);
I;
J := ideal(4*x, x^3, 2*y,3*z);
I*J;
I+J;
IsZero(I);
x*I;
I*y;
(2*x/x)*I;
LT(I);     --> EXPECT ERROR
GBasis(J); --> EXPECT ERROR
radical(I);--> EXPECT ERROR
intersection(I,J); --> EXPECT ERROR
```

Currently (2021-04-28) all tests behave as hoped.

#10 - 28 Apr 2021 14:44 - John Abbott

I have just checked in my current changes... they are not well designed (but the test above works as hoped).

#11 - 16 Feb 2024 09:47 - John Abbott

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