CoCoALib - Feature #658

Indets actually in a poly (or vector or matrix)

22 Jan 2015 15:43 - John Abbott

Status:	Closed	**Start date:**	22 Jan 2015
Priority:	Urgent	**Due date:**	
Assignee:	John Abbott	**% Done:**	100%
Category:	New Function	**Estimated time:**	4.66 hours
Target version:	CoCoALib-0.99700	**Spent time:**	4.60 hours

Description

Should we have a function which says which indets actually appear in a polynomial?

Related issues:

Related to CoCoALib - Feature #562: Subrings	**New**	**20 May 2014**
Related to CoCoA-5 - Bug #100: BringIn should map only the indets in its arg	**New**	**07 Mar 2012**
Related to CoCoA-5 - Slug #687: Builtin fn makes unnecessary copy of arg	**Closed**	**19 Apr 2015**
Related to CoCoALib - Feature #91: Return type & name for "indets" of a PP	**Closed**	**11 Feb 2012**
Related to CoCoALib - Feature #1103: Pseudo-zero-dim ideals	**In Progress**	**19 Sep 2017**
Related to CoCoALib - Feature #1349: ideal ctor where given gens are a gbasis	**In Progress**	**24 Oct 2019**

History

#1 - 22 Jan 2015 15:46 - John Abbott

If the answer is **yes** then we must decide some things:

- what is the answer? List of polys which are actually indets? List of indices?
- should "indets" in the coeff ring be considered?
- to what types can it be applied? RINGELEM, LIST, IDEAL??, MATRIX

What should it be called? indets is already defined for rings; we could extend that, or else use a new name.

#2 - 22 Jan 2015 19:08 - Anna Maria Bigatti

John Abbott wrote:

> If the answer is **yes** then we must decide some things:
>
> - what is the answer? List of polys which are actually indets? List of indices?

in CoCoALib it should be the list of indices, in CoCoA-5 I'm not sure...

> - should "indets" in the coeff ring be considered?

no

> - to what types can it be applied? RINGELEM, LIST, IDEAL??, MATRIX

RINGELEM yes
IDEAL no (generator dependant)
LIST inclined for no (elements might be in different rings!)

MATRIX maybe

> What should it be called? indets is already defined for rings; we could extend that, or else use a new name.

indets would be ambiguous, I believe. Said that, I cannot think of another good name ;-)

#3 - 14 Apr 2015 13:59 - John Abbott

A possible name: **IndetsIn**

I would like to have **IndetsIn** also for a list (which is assumed to be "homogeneous", *i.e.* all elements in the same poly ring o/w error). I used it in some CoCoA-5 code for checking whether a set of conditions defines an arithmetic group (these conditions would then become generators of an ideal).

#4 - 14 Apr 2015 14:05 - John Abbott

Here are two potentially related fns:

- InvolvesOnlyTheseIndets which returns a bool -- could be much faster than checking IndetsIn for being a subset
- PolyInIndetPower which returns a list of PPs such that the arg is a polynomial in these PPs (*e.g.* given x^10+x^5+2 the result would be x^5)

The second fn is potentially quite complicated, but could be useful for mapping some multivariate polynomials down to simpler ones. Hmmm, technically a homogeneous bivariate polynomial is "equivalent" to a univariate polynomial, but it would not fit this scheme... :-/

NOTE we can consider InvolvesOnlyTheseIndets if some genuine applications for it arise.

NOTE2 I am not sure what semantics the fn PolyInIndetPower should have; it could certainly be useful in GCD and factorization...

#5 - 17 Apr 2015 18:32 - John Abbott

I have impl'ed a first version (simple rather than efficient).
Not yet checked in; I want to do some testing first.

#6 - 19 Apr 2015 18:17 - John Abbott

I am still undecided about the "return" value (and its type):

- **A** a list of integer indices (presumably of type vector<long>)
- **B** a vector "bools" with k-th posn being set to "true" if k-th indet is present

A possible advantage of **B** is that it would allow checking to finish early if all entries are "true" (*e.g.* when checking a list of polynomials, checking could end part way through a poly even if that poly does not contain all indets). Not sure how genuinely useful this could be.

We could even offer both interfaces (but that seems excessive for a seemingly unimportant function).

In **B** the return type vector<bool> is probably a poor choice because I suspect that setting individual bits is likely to be costly; vector<int> should be faster -- I'll check! **ANSWER** [in IndetsIn in SparsePolyRing.C:965] there's very little difference (hardly measurable), so vector<bool> is a better choice because it more clearly expresses the purpose.

#7 - 20 Apr 2015 10:26 - Anna Maria Bigatti

also related to #91

#8 - 13 May 2015 17:59 - John Abbott

An advantage of vector<bool> is that it avoids the problem/incompatibility of indices in C++ starting from 0 (both for vectors and indets) whereas in CoCoA-5 they start from 1.

#9 - 29 Jun 2015 17:23 - John Abbott

- Status changed from New to In Progress

- % Done changed from 0 to 10

Anna suggests the result could be a DynamicBitset; these appear to be like vector<bool> but also support bitwise logic operations.

#10 - 28 Nov 2016 22:21 - John Abbott

- Project changed from CoCoA-5 to CoCoALib

- Subject changed from Indets actually in a poly to Indets actually in a poly (or vector or matrix)

- Category changed from CoCoA-5 function: new to New Function

- Target version changed from CoCoA-5.?.? to CoCoALib-1.0

I have moved this issue to CoCoALib, which is where I had expected to find it.

On several occasions I would have liked to have a function which tells me "which subring" the computation is in (*e.g.* maybe I simply needed to pick an indet from those actually appearing in the polynomial).

#11 - 16 Jan 2018 18:16 - John Abbott

This is definitely needed: **There are awkward work-arounds in radical.cpkg5 and in BringIn.cpkg5.**

#12 - 16 Jan 2018 18:17 - John Abbott

- Related to Feature #1103: Pseudo-zero-dim ideals added

#13 - 17 Jan 2018 08:38 - Anna Maria Bigatti

- Target version changed from CoCoALib-1.0 to CoCoALib-0.99600

#14 - 12 Jun 2018 18:30 - John Abbott

- Target version changed from CoCoALib-0.99600 to CoCoALib-0.99650 November 2019

#15 - 04 Apr 2019 21:31 - John Abbott

- Priority changed from Normal to High

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

- % Done changed from 10 to 20

It seems that there is already an impl in SparsePolyOps-RingElem.

There is no doc, no test. Also not exported to CoCoA-5.

It would be nice to finish this soon (but there are still some questions to answer, see above).

Another possible return value is a product of the indets actually appearing. This has the feature that it retains the identity of the poly ring even if there are no indets (since the result is then one(P)).

#16 - 24 Oct 2019 15:00 - John Abbott

- % Done changed from 20 to 30

I have just added (in SparsePolyOps-RingElem.C) an impl for IndetsIn(const std::vector<RingElem>&).
The result is vector<long> the same as what the existing impl of IndetsIn(ConstRefRingElem) produces.

I'm now trying to use it to modify MinPolyXYZ so that it works with "relatively zero-dim ideals". Not entirely trivial :-/

PS the new impl uses the "new" C++11 for loop syntax (and a goto statement)

#17 - 24 Oct 2019 16:26 - John Abbott

- Related to Feature #1349: ideal ctor where given gens are a gbasis added

#18 - 29 Oct 2019 12:37 - John Abbott

Checked in IndetsIn(vector<RingElem>)

#19 - 25 Jan 2020 15:26 - John Abbott

- Status changed from In Progress to Resolved

- Assignee set to John Abbott

- % Done changed from 30 to 70

- Estimated time set to 3.99 h

Comment 18 about says that I have checked the code in (seems to be true!)

Still no doc; and probably no examples.

#20 - 13 Feb 2020 14:24 - John Abbott

- Priority changed from High to Urgent

After discussing with Anna, she wants the result to be a polynomial (which is a 1-monomial) whose value is the product of the indets appearing in the polynomial(s).

What should the name of this version be?
JAA thinks something of the form **IndetsXYZ**; but should XYZ be? A possibility is **IndetSupport**; in that "support" of any sparse structure is the "list" of the parts actually appearing. Anna thinks **IndetsIn** might still be OK. Anna also suggested **IndetsProd**; this is accurate, but it is hard to guess what the fn does just from its name.

The version proposed by Anna produces a value which is the radical of the product of the supports, or the radical of the "top" (lcm) of the supports.
But it seems hard to produce a reasonably short name which is comprehensible (and captures this fact).

#21 - 13 Feb 2020 15:09 - John Abbott

- Status changed from Resolved to Closed

- % Done changed from 70 to 100

- Estimated time changed from 3.99 h to 4.66 h

Implemented **IndetsProd** -- advantage is that it gives same interface in CoCoALib as in CoCoA-5. **IndetsIn** is also available in CoCoALib.

Added doc, example, test(?)