CoCoALib - Feature #1694

New expv/exponents function?

09 Aug 2022 11:57 - John Abbott

Status:	Closed	**Start date:**	09 Aug 2022
Priority:	Normal	**Due date:**	
Assignee:	John Abbott	**% Done:**	100%
Category:	New Function	**Estimated time:**	3.33 hours
Target version:	CoCoALib-0.99850	**Spent time:**	3.35 hours

Description

Consider whether to implement a simpler fn for computing the expv of a PP.

Currently one must do:

```
vector<long> expv;
...
exponents(expv, PP);
// now use expv e.g. in a call to PushBack
```

It would be neater to be able to write simply expv(PP);
for instance th hypothetical call to PushBack would become
just a single line something like

```
PushBack(G, coeff, expv(PP));
```

The disadvantage is that each call to expv entails creating a vector<long>,
but this inefficency may be unimportant in some contexts (and it may be more
important to write clear code).

Discuss!

History

#1 - 09 Aug 2022 12:06 - John Abbott

I have been discussing with Bruns over email. He wanted to write something like

```
PushBack(G, coeff, PP(iter));
```

where iter is a SparsePolyIter.

This code threw an exception (because the PPMonoid of the ring of G was not the same as that of PP(iter)).
After some discussion Bruns preferred creating the expv and then using that because *the code is more robust*
(*i.e.* does not depend on the PPMonoids being identical).

In other words, CoCoALib's user interface is forcing him to write "ugly" code. The design decision behind
the current interface was to avoid unnecessary (run-time) inefficiency [but has this even been measured?]

#2 - 09 Aug 2022 12:14 - John Abbott

Another way to solve Bruns's problem would be to allow PushBack to accept PPs from a different monoid, and then map them automatically (via expv presumably) into the correct monoid.

This would give a simple/clean/uniform interface. What would not be clear is how fast the call would be: if the PPMonoids are identical, it will be fast; otherwise it will be slower since it has to allocate and compute the expv internally.

If we do adopt this idea, do we require that the number of indets in the two PPMonoids is the same? Or is it OK if the destination PPMonoid has more indets?

Qn Or even if the destination PPMonoid has fewer indets we could map just the first indets? If there are unmapped indets, do we get an exception? Or are they silently ignored?

Ans I think it may be better to handle the "non-identity case" via an explicit PPMonoid homomorphism. Then it would be clear to a reader of the code that something non-trivial is going on.

#3 - 22 Aug 2022 15:34 - John Abbott

- Status changed from New to In Progress

- % Done changed from 0 to 10

Bruns replied by email (2022-08-09):

```
I am not sure whether the code is ugly. But I would implement
the PP version by having it call it  the exponent vector version.

For the exponent vector, I think a function returning the vector
(instead of giving it back through an argument) is more natural.
```

#4 - 22 Aug 2022 16:01 - John Abbott

- Assignee set to John Abbott

- % Done changed from 10 to 50

After discussing with Anna, we think the following redesign is a good solution:

- change existing exponents so that it returns **const std::vector<long>&** instead of void (where the reference is to the vector passed in as workspace)
- add a new fn (also called exponents?) which takes just the PP, and returns a newly allocated vector of the exponents (by value, not by

reference)

The new fn is potentially less efficient inside a loop (since it repeatedly does alloc-free)

#5 - 22 Aug 2022 17:29 - John Abbott

- Status changed from In Progress to Resolved

- % Done changed from 50 to 70

I have implemented the changes proposed in comment 4.
It all compiles, and all tests pass.
I'll check in soon.

#6 - 14 Sep 2022 21:25 - John Abbott

- % Done changed from 70 to 80

I have checked in.
It remains to consider and maybe update every call to exponents (or BigExponents)

#7 - 07 Mar 2023 20:54 - John Abbott

Let's discuss this issue. I hope we can close it this week!

#8 - 09 Mar 2023 21:48 - John Abbott

- Status changed from Resolved to Closed

- % Done changed from 80 to 100

- Estimated time set to 3.33 h

I have cleaned up all calls which I think needed cleaning.

Observation: I thought it was odd that exponents(expv, PP) returns a **const** result, but then decided that it is probably safer so.