CoCoALib - Feature #156

Brand new symbol(s)

07 May 2012 16:46 - John Abbott

Status:	Closed	**Start date:**	07 May 2012
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	100%
Category:	New Function	**Estimated time:**	25.00 hours
Target version:	CoCoALib-0.9951	**Spent time:**	8.40 hours

Description

The pseudo-ctors for PPMonoids all require that names be specified for the indets.

It can occur that we want to create a new polynomial extension over a user supplied ring (which may contain any number of symbols with any names). To avoid problems with name clashes between symbols, it will be useful to have a means of generating a brand new symbol, guaranteed to be different from (& compatible with) all previsouly created symbols. JAA recalls that REDUCE had a similar feature called "gensym".

Such a function could be called by PPMonoid pseudo-ctors which do not ask for names for the indets.

Related issues:

Related to CoCoALib - Bug #155: gcd: multivariate over non-prime finite field	**Closed**	**07 May 2012**
Related to CoCoALib - Feature #157: Separate ThreadsafeCounter from symbol.C	**Closed**	**10 May 2012**

History

#1 - 07 May 2012 16:54 - John Abbott

JAA's first suggestion is to create "gensyms" containing a single index (in the hope that the index will not overflow). I was not thinking of recycling used gensyms. The corresponding pseudo-ctor must be added to **symbol.H**

If we follow this suggestion then we need to choose the head of the symbol. One possibility is "?" -- JAA thinks that Maple used to adopt this convention. JAA prefers to keep the head short (ideally just 1 or 2 chars long). The head should be one which a normal user cannot choose; compatibility with other symbols having the same head is guaranteed, because all symbols with that head have one index and all newly created symbols have a higher index than previously created symbols.

There must be a **global** counter for these gensyms -- requires locking in multithreaded environments, but this should not cause too much trouble as I doubt any program will create vast numbers of gensyms.

#2 - 07 May 2012 16:58 - John Abbott

Here are 3 proposals for new PPMonoid pseudo-ctor signatures:

```
PPMonoid NewPPMonoid(const PPOrdering& ord);
PPMonoid NewPPMonoid(const MachineInt& nvars, const PPOrdering& ord);
PPMonoid NewPPMonoid(const MachineInt& nvars, const PPOrderingCtor& ord);
```

#3 - 07 May 2012 17:01 - John Abbott

This is hardly important, but might be beneficial in a multithreaded environment. Possibly it would be "cleaner" too.

We could also have a function that creates a list of new symbols. The advantage is that it could increment the global counter just once by the appropriate amount instead of repeatedly incrementing it by one (with multiple locking/unlocking manoeuvres).

#4 - 07 May 2012 17:15 - John Abbott

- Estimated time set to 25.00 h

Here are some more proposals for the head: **??** or **#** or **_**

Printed value would look like: **3*??[17]^2-2** or **3*#[17]^2-2** or **3*_[17]^2-2** or **3*?[17]^2-2**

Note that my earlier suggestion of just **?** as the prefix could lead to ambiguity with my proposal for the "Iverson" bracket.

Right now I have a preference for **#**. The character **#** has no other meaning; it is fairly visible inside a formula without being unpleasantly obtrusive.

Under normal circumstances I would not expect "unnamed" symbols to be printed, but they must be printable (if only for debugging).

#5 - 07 May 2012 18:43 - Anna Maria Bigatti

John Abbott wrote:

> Here are some more proposals for the head: **??** or **#** or **_**
>
> (...)
>
> Right now I have a preference for **#**. The character **#** has no other meaning; it is fairly visible inside a formula without being unpleasantly obtrusive.
>
> Under normal circumstances I would not expect "unnamed" symbols to be printed, but they must be printable (if only for debugging).

I vote for **#**

#6 - 08 May 2012 09:49 - John Abbott

JAA suggests that internally the head of an "anonymous" symbol be the empty string; the accessor function then checks the internal repr head, and if it is empty it returns whatever string we choose. An advantage of using empty strings internally is that we do not waste space allocating lots of copies of our chosen "anonymous head".

I have already checked that empty heads are forbidden for normal symbols.

#7 - 09 May 2012 17:18 - John Abbott

The current suggestion for imlpementation is the following:

- internally an anonymous symbol has empty head
- the printing function prints out whatever "fake" head we choose
- the **head** function returns an empty string
- probably add a new fn **IsAnonymous** (name still under consideration)

An advantage of having **head** return an empty string is that any user code which checks the head will not need changing if we change the convention for the "fake" head.

The only function which actually depends on the choice of fake head is the printing function.

#8 - 10 May 2012 15:53 - John Abbott

- Status changed from New to In Progress

- % Done changed from 0 to 20

JAA has completed a first impl. The new public fns (pseudo-ctors) are:

- **NewSymbol()** which creates a single new nameless symbol
- **NewSymbols(n)** which creates a vector containing **n** new nameless symbols (with sequential indices)

These implementations are threadsafe (using features from BOOST threads library).

A new class **ThreadsafeCounter** has been put in symbol.C; it will be separated later.

#9 - 10 May 2012 16:38 - John Abbott

I have opted not to implement **IsAnonymous** because I cannot think of when it might actually be useful.

The current impl is such that **head** applied to an anonymous symbol will give the empty string.

Added doc and a new example for the new fns.

#10 - 10 May 2012 16:48 - John Abbott

Check through existing code, and use anonymous symbols where appropriate. Then check everything!

#11 - 10 May 2012 17:16 - John Abbott

It is simpler to move the "special" ctor into the private zone, and remove the **AnonymousMarker**. This should take just a few mins.

#12 - 25 May 2012 15:15 - Anna Maria Bigatti

- Category set to New Function

- Target version set to CoCoALib-0.9951

#13 - 28 May 2012 16:13 - John Abbott

- Status changed from In Progress to Closed

- % Done changed from 20 to 100

I've cleaned the code, and updated the documentation.

Note: Task #157 will make a further change to this one.

Anna has convinced me that any pseudo ctor for creating a PolyRing where the user does not specify names for the indets should use normal names (*i.e.* x[N]) rather than anonymous symbols. So the existing implementations are fine.