

CoCoA-5 - Bug #946

Function "ideal" evaluates the argument twice

17 Oct 2016 11:03 - Anna Maria Bigatti

Status:	Resolved	Start date:	17 Oct 2016
Priority:	Urgent	Due date:	
Assignee:	Anna Maria Bigatti	% Done:	80%
Category:	Parser/Interpreter	Estimated time:	6.00 hours
Target version:	CoCoA-5.4.2	Spent time:	13.20 hours
Description			
ideal(RING, func(...)) evaluates func once, ideal(func(...)) evaluates func twice.			
Related issues:			
Related to CoCoA-5 - Slug #31: theValue makes copy			In Progress 15 Nov 2011

History

#1 - 17 Oct 2016 11:36 - John Abbott

- Status changed from New to In Progress
- Priority changed from High to Urgent
- % Done changed from 0 to 10

Having just looked at the code, I'm not entirely surprised.

Also why are there calls to evalArgAsListOfRingElem and evalArgAsListOf<RingElem>? :-/

It seems to me that the arg has already been evaluated in the call to evalArgAsT1orT2orT3, and saved in the variable x.

#2 - 17 Oct 2016 11:40 - John Abbott

Two test cases:

```
define f(x) println "Inside f"; return x; enddefine;
I1 := ideal(f(x)); --> BAD
I2 := ideal([f(x)]); --> BAD
I3 := ideal(R, [f(x)]);--> OK
```

#3 - 17 Oct 2016 12:18 - John Abbott

Here is some code we could add to one of the CoCoA-5 tests:

```
use R ::= QQ[x];
define f(x)
  TopLevel FLAG;
  if FLAG then error("Called f twice"); endif;
  FLAG := true;
  return x;
enddefine; -- f
```

```

FLAG := false;
I1 := ideal(f(x));

FLAG := false;
I2 := ideal([f(x)]);

FLAG := false;
I3 := ideal(R, [f(x)]);

```

Probably the name of the top-level variable should be a bit longer (and more unusual).
The test prints nothing if all is well, otherwise a CoCoA error is reported.

#4 - 17 Oct 2016 12:39 - John Abbott

Are there any other fns with a similar problem? 8-{

#5 - 17 Oct 2016 13:13 - John Abbott

Oh dear! The problem is more widespread. Even len evaluates its arg twice 8-O

#6 - 17 Oct 2016 17:41 - John Abbott

From looking at the code in BuiltinFunctions-CoCoALib.C, here is a list of other suspect impls:

- **len**
- **syz** (of a list)
- **SyzOfGens** just what is this code supposed to do???
- **homog** where first arg is a list
- **gcd** of a list of int or list of ring elem
- **lcm** (like gcd)
- **apply** where 2nd arg is a list
- **ContentWRT** where 2nd arg is a list
- **CoefficientsWRT** where 2nd arg is a list
- **ElimMat** where 1st is a list, or where 2nd arg is an INT
- **ideal**
- **NewFreeModuleForSyz** where arg is a list
- **submodule** where 1st arg is a list
- **elim** where 1st arg is a list
- **PolyAlgebraHom** where 3rd arg is a list
- **PolyRingHom** where 4rd arg is a list

JAA thinks these are all the cases from BuiltinFunctions-CoCoALib.C

#7 - 09 Mar 2017 17:18 - John Abbott

For an **urgent** issue, this is making very slow progress :-)

Perhaps someone (who??) should produce a test file which covers all the fns listed in comment 6 above. Then someone (who??) can try cleaning up the code... any volunteers?

#8 - 09 Mar 2017 18:01 - Anna Maria Bigatti

I think this is far too difficult for this release.
I suggest moving it to next.

#9 - 10 Mar 2017 13:06 - John Abbott

- *Target version changed from CoCoA-5.2.0 spring 2017 to CoCoA-5.2.2*

Despite the **urgent** status I have postponed this to the next version. After all it is principally a matter of efficiency rather than correctness. Moreover, a simple workaround should be just to put values in variables and pass the variables to the affected functions: evaluating a variable twice (rather than just once) should be barely noticeable.

#10 - 16 May 2017 18:07 - Anna Maria Bigatti

- *Status changed from In Progress to Resolved*
- *Assignee set to Anna Maria Bigatti*
- *Estimated time changed from 3.00 h to 6.00 h*

Found it! (and fixed most of them)

The problem is when using the hand functions `EvalArgAsT1OrT2...<...>(ARG(0))`, which): this returns a `RightValue` which is then (usually) converted with `RefTo<class>`.

Unfortunately sometimes it is not as simple as this and `EvalArgAs<...>(ARG(0))` is called again on the same argument, and, as the name says, `EvalArgAs..` evaluates the argument again.

One common case is for a list of `RingElem`, so I now implemented `evalRVAsListOfRingElem(RightVal, ARG(0))` (RV is for `RightValue`) which takes the already processed `RightVal`, and ARG only for indicating errors, in case.

For other functions I just made sure ARG is not used twice, but there are still a few cases (a funny case in `gcd`, for example)

#11 - 16 May 2017 18:16 - Anna Maria Bigatti

- *% Done changed from 10 to 70*

#12 - 17 May 2017 08:30 - Anna Maria Bigatti

I made a test: "bug-EvalTwice.cocoa5" where we can collect all cases we meet, solved or unsolved.

#13 - 15 Dec 2017 15:30 - John Abbott

- *Target version changed from CoCoA-5.2.2 to CoCoA-5.2.4*

Postponing to next version; it is better to deal with all fns, and then close (rather than close after having dealt with just some of the fns).

#14 - 30 Jul 2018 14:33 - John Abbott

- Target version changed from CoCoA-5.2.4 to CoCoA-5.3.0

#15 - 02 Oct 2019 16:19 - John Abbott

- Target version changed from CoCoA-5.3.0 to CoCoA-5.4.0

#16 - 16 Oct 2019 22:30 - John Abbott

- Target version changed from CoCoA-5.4.0 to CoCoA-5.3.0

I have just run the test given in comment 2; now only the first one evals its arg twice.
Is this easy to fix?

Moving target back to 5.3.0. We should at least produce a test suite for the fns listed in comment 6, so we know how much progress we are making (and how far we are from finishing). If the test suite shows that we are far from finishing then we can change the target back to 5.3.2.

#17 - 13 Feb 2020 16:03 - John Abbott

- Target version changed from CoCoA-5.3.0 to CoCoA-5.4.0

#18 - 13 Feb 2020 16:04 - John Abbott

Maybe the move ctor from C++14 can help here?
Postponed because it will take (a lot of) time to fix.

#19 - 16 Feb 2021 14:41 - John Abbott

I might have fixed this (for ideals)... and probably introduced some new weird bugs.... the interpreter code is not easy to fathom :-(

#20 - 17 Feb 2021 19:03 - John Abbott

I have just checked in my revised code (even though it still needs to be cleaned).

We must develop a proper test suite for the various cases listed in comment 6. This will take some time; I hope most cases have already been resolved, but an objective test would give a nice confirmation.

#21 - 10 Jun 2021 19:22 - Anna Maria Bigatti

We must develop a proper test suite for the various cases listed in comment 6. This will take some time; I hope most cases have already been resolved, but an objective test would give a nice confirmation.

See [#946-12](#)

"I made a test: "bug-EvalTwice.cocoa5" where we can collect all cases we meet, solved or unsolved."

#22 - 02 Jul 2021 16:17 - Anna Maria Bigatti

at the end of this file there is a section "TO BE FIXED" with the last few cases to be fixed.

Anna Maria Bigatti wrote:

We must develop a proper test suite for the various cases listed in comment 6. This will take some time; I hope most cases have already been resolved, but an objective test would give a nice confirmation.

See [#946-12](#)

"I made a test: "bug-EvalTwice.cocoa5" where we can collect all cases we meet, solved or unsolved."

#23 - 02 Jul 2021 16:20 - John Abbott

Created new fn **SpecializeToListOfRingElem** which converts a generic LIST into a list of RINGELEM (I hope).
See Interpreter.H

#24 - 02 Jul 2021 17:27 - John Abbott

- % Done changed from 70 to 80

Just 1 left: ModuleElem

Relevant source seems to be in BuiltinFunctions-CoCoALib.C around line 1465.
At the moment, I have no idea why the arg is evaluated twice... the source code looks sane.

#25 - 19 Jul 2021 14:13 - Anna Maria Bigatti

(I'll see ModuleElem)

#26 - 30 Jul 2021 11:13 - Anna Maria Bigatti

Anna Maria Bigatti wrote:

(I'll see ModuleElem)

Worked together: ModuleElem is fine. The problem is in submodule (see ideal)

#27 - 30 Jul 2021 12:16 - John Abbott

Should check all fns which call **evalArgAsT1or**
My computer/fgrep says there are 85 such fns.

#28 - 30 Jul 2021 17:32 - Anna Maria Bigatti

Fixed PolyRingHom and PolyAlgebraHom.
Added evalRVAsListOfRingElem with ring
CVS'd

#29 - 03 Feb 2022 19:32 - John Abbott

Can we close this issue? If not, postpone it!

#30 - 09 Feb 2022 14:26 - Anna Maria Bigatti

fixed **evalArgAsListOfSymbols**, updated test (NewPolyRing)

#31 - 23 Feb 2022 12:14 - Anna Maria Bigatti

- *Target version changed from CoCoA-5.4.0 to CoCoA-5.4.2*

Mostly done, enough for CoCoA-5.4.0.

Still to be fixed submodule (see bug-EvalTwice.cocoa5).
Probably it is the last one missing.

#32 - 14 Mar 2023 22:12 - John Abbott

- *Related to Slug #31: theValue makes copy added*