

## CoCoALib - Bug #971

### CheckForInterrupt does not work in the expected way

14 Nov 2016 11:10 - John Abbott

<b>Status:</b>	Closed	<b>Start date:</b>	14 Nov 2016
<b>Priority:</b>	Normal	<b>Due date:</b>	
<b>Assignee:</b>	John Abbott	<b>% Done:</b>	100%
<b>Category:</b>	Improving	<b>Estimated time:</b>	3.01 hours
<b>Target version:</b>	CoCoALib-0.99550 spring 2017	<b>Spent time:</b>	3.25 hours
<b>Description</b>			
Looking at the call to CheckForInterrupt in GReductor::myReduceCurrentSPoly (around line 760 of TmpGReductor.C), I see that Anna had to do something strange to make it work.			
Investigate, and arrange for a simple call to CheckForInterrupt to suffice.			
<b>Related issues:</b>			
Related to CoCoALib - Feature #714: Interrupt mechanism		<b>Closed</b>	<b>19 May 2015</b>
Related to CoCoA-5 - Feature #744: Handle interrupts more helpfully		<b>Closed</b>	<b>01 Jul 2015</b>
Related to CoCoALib - Design #982: Catching interrupts in example progs?		<b>Closed</b>	<b>25 Nov 2016</b>
Related to CoCoALib - Bug #1458: Redesign interrupt mechanism?		<b>Rejected</b>	<b>10 May 2020</b>

### History

#### #1 - 14 Nov 2016 11:12 - John Abbott

- Related to Feature #714: Interrupt mechanism added

#### #2 - 14 Nov 2016 13:03 - John Abbott

- Status changed from New to In Progress

- % Done changed from 0 to 10

As far as I can see, the CoCoA-5 interpreter needs to be modified so that it catches both CoCoA::ErrorInfo and CoCoA::InterruptReceived objects. At the moment it seems to ignore the latter... to be honest I am not even sure how it achieves the current behaviour.

There are 11 places where Interpreter.C catches ErrorInfo -- why so many???

Do I need to add another catch command (for InterruptReceived) after each one?

The other problem is what to do once I have caught an InterruptReceived. With an ErrorInfo the existing code simply creates a RuntimeException object (incl. information about where in the CoCoA-5 source the interpreter had reached).

In amongst the CoCoA-5 code, I see that there is also InterruptException; this is presumably the correct one to throw, but in its current form the sole constructor is very limiting. I had better add some more ctors... sigh!

#### #3 - 14 Nov 2016 13:07 - John Abbott

In issue [#714](#) I had guessed that the only place I needed to catch InterruptReceived was around line 2678 of Interpreter.C.

I now report that **this does not suffice**. Since CoCoALib watches for interrupts in RingBase::mySequentialPower, I tried computing the 1000th power of a polynomial, and then send an interrupt during the computation. The interpreter did stop when I interrupted the computation, but it gave no message such as "Interrupt received". This happened only when I added the appropriate catch statement after line 1415.

#### #4 - 14 Nov 2016 16:11 - John Abbott

- Assignee set to John Abbott

- % Done changed from 10 to 20

I have made some changes to Interpreter.C (mostly adding catch commands for InterruptReceived).

I have not (yet) changed the way Anna used CheckForInterrupt in TmpGReductor.C; while its use in RingBase::mySequentialPower is as I had intended. The following transcript show the slightly different behaviour which CoCoA-5 exhibits when CoCoALib is interrupted:

```
>>> f := x+y+z;
>>> g := f^300; // START THIS THEN INTERRUPT IT!
C-c C-c
-----
>>> CoCoA interrupted <<<
-----

--> ERROR:

*** Interrupted ***

--> g := f^300; // START THIS THEN INTE ...
-->      ^

>>> S := support(f^30);
>>> Sshifted := subst(S, [[x,x-2],[y,y+3],[z,z-5]]);
>>> I := ideal(Sshifted); // strangely slow!
>>> GB := GBasis(I); // START THIS THEN INTERRUPT IT!
C-c C-c
-----
>>> CoCoA interrupted <<<
-----

--> ERROR: InterruptReceived
--> GB := GBasis(I); // START THIS THEN INTERRUP ...
-->      ^^^^^^^^^^

>>>
```

Comments? Opinions? Preferences?

## #5 - 14 Nov 2016 21:53 - Anna Maria Bigatti

Cane we have something more compact like this?

```
>>> f := x+y+z;
>>> g := f^300; // START THIS THEN INTERRUPT IT!
C-c C-c
-----
>>> CoCoA interrupted <<<
-----

--> ERROR:   *** Interrupted ***
--> g := f^300; // START THIS THEN INTE ...
-->      ^
```

## #6 - 18 Nov 2016 20:52 - John Abbott

- Related to Feature #744: Handle interrupts more helpfully added

## #7 - 18 Nov 2016 21:58 - John Abbott

- % Done changed from 20 to 50

I have now changed the call to CheckForInterrupt in TmpGReductor to the simple call that I had expected to see, and it works as I hoped/intended/expected/wanted/etc.

Since it is now just a matter of inserting calls to CheckForInterrupt("Fn name"); I am hoping that soon several new calls will be judiciously inserted so that lengthy CoCoALib computations can be interrupted with only a reasonable wait for recognition of the interruption.

## #8 - 25 Nov 2016 17:30 - John Abbott

- Status changed from In Progress to Feedback

- % Done changed from 50 to 90

## #9 - 25 Nov 2016 17:49 - John Abbott

- Related to Design #982: Catching interrupts in example progs? added

## #10 - 29 Mar 2017 18:09 - John Abbott

- Status changed from Feedback to Closed

- Target version changed from CoCoALib-0.99560 to CoCoALib-0.99550 spring 2017

- % Done changed from 90 to 100

## #11 - 28 Apr 2017 09:30 - Anna Maria Bigatti

- Estimated time set to 3.01 h

## #12 - 10 May 2020 11:54 - John Abbott

- Related to Feature #1457: Make SmoothFactor interruptible added

**#13 - 10 May 2020 11:55 - John Abbott**

- Related to deleted (Feature #1457: Make SmoothFactor interruptible)

**#14 - 10 May 2020 12:06 - John Abbott**

- Related to Bug #1458: Redesign interrupt mechanism? added