CoCoALib - Feature #718

Insert calls to CheckForInterrupt

21 May 2015 10:52 - John Abbott

Status:	Resolved	**Start date:**	21 May 2015
Priority:	Normal	**Due date:**	
Assignee:	Anna Maria Bigatti	**% Done:**	90%
Category:	Improving	**Estimated time:**	0.00 hour
Target version:	CoCoALib-0.99880	**Spent time:**	2.10 hours

Description

So that interrupts are acted upon we must insert calls to CheckForInterrupt in various places -- principally in loops which may take a long time, I believe.

Related issues:

Related to CoCoALib - Feature #714: Interrupt mechanism	**Closed**	**19 May 2015**
Related to CoCoALib - Feature #721: CheckForInterrupt: string arg to specify ...	**Closed**	**29 May 2015**
Related to CoCoALib - Feature #638: Time limit: let user specify time limit f...	**Closed**	**27 Oct 2014**
Related to CoCoA-5 - Feature #744: Handle interrupts more helpfully	**Closed**	**01 Jul 2015**
Related to CoCoA-5 - Support #1023: CoCoAManual for verbosity: how to find wh...	**Closed**	**06 Mar 2017**
Related to CoCoALib - Feature #1457: Make SmoothFactor interruptible	**Closed**	**10 May 2020**
Related to CoCoALib - Feature #1633: Make polynomial multiplication interrupt...	**Closed**	**16 Nov 2021**
Related to CoCoALib - Feature #127: Convert DUPFF code to C++	**In Progress**	**05 Apr 2012**

History

#1 - 21 May 2015 10:54 - John Abbott

I suggest we keep a log here of the files we have looked at with a view to inserting calls to CheckForInterrupt.

#2 - 29 May 2015 12:03 - Anna Maria Bigatti

- Status changed from New to In Progress

- Assignee set to Anna Maria Bigatti

- % Done changed from 0 to 10

Nice! I added it into TmpGReductor.C (I was actually doing something else with very slow GB ;-))

```
  void GReductor::myReduceCurrentSPoly()
  {
    try { CheckForInterrupt(); } // C-c for long computations
    catch (const InterruptReceived&)
    { CoCoA_ERROR("InterruptReceived", "GReductor::myReduceCurrentSPoly"); }
.....
```

I'm not quite sure it is what intended, but it works.

#3 - 08 Jun 2015 11:09 - John Abbott

Mario asked about the possible run-time cost of calling CheckForInterrupt.

Currently it may be needlessly expensive because the function InterruptFlag() defined in GlobalManager.H does several checks:

1. check that GlobalManager has been created
2. check that a flag has been specified (*i.e.* if a pointer is non-NULL)
3. get the value of the pointed to flag.

It may be simpler to define a global flag belonging to CoCoALib (but which is simply left as "false"). I think we could also skip checking whether GlobalManager has been created -- in the worst case you'll get a SEGV when calling CheckForInterrupt -- but I'd be amazed if that was the first time GlobalManager was needed. Perhaps I'll do a quick overhead check for the current implementation.

#4 - 02 Feb 2016 14:14 - John Abbott

- Target version changed from CoCoALib-0.99540 Feb 2016 to CoCoALib-1.0

#5 - 17 Feb 2016 11:38 - John Abbott

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

#6 - 10 Nov 2016 17:51 - John Abbott

I tried interrupting a long gbasis computation (as part of one of Mario's long radical examples). The computation had been running for about 50 hours (probably not all the time in that gbasis computation), but the interrupt took over 15 minutes to be recognized. That is a bit slow!

It is quite possible that CoCoALib was doing a reduction of a "hideously large" polynomial; I'm not sure whether it is really worth trying to interrupt inside a reduction... :-/

#7 - 07 Apr 2017 11:40 - Anna Maria Bigatti

- Related to Support #1023: CoCoAManual for verbosity: how to find which levels? how to find which functions? added

#8 - 10 May 2020 11:54 - John Abbott

- Related to Feature #1457: Make SmoothFactor interruptible added

#9 - 16 Nov 2021 20:27 - John Abbott

- Target version changed from CoCoALib-1.0 to CoCoALib-0.99850

- % Done changed from 10 to 60

#10 - 16 Nov 2021 20:34 - John Abbott

- Related to Feature #1633: Make polynomial multiplication interruptible? added

#11 - 08 Aug 2022 19:56 - John Abbott

Nico Mexis asked whether there could be a check for interrupts inside the factorizer.
This will much easier to achieve when the factorizer is modernized... if ever (sigh).

#12 - 28 Sep 2022 15:04 - John Abbott

- Related to Feature #127: Convert DUPFF code to C++ added

#13 - 10 Mar 2023 17:48 - John Abbott

- Status changed from In Progress to Resolved

- Target version changed from CoCoALib-0.99850 to CoCoALib-0.99880

- % Done changed from 60 to 90

I do this whenever I encounter a case which is not yet covered.
I think most cases are now covered. Hence 90%. But postponing as well.

#14 - 23 Oct 2023 12:05 - John Abbott

Make matrix **inverse** interruptible (at least for matrices over ZZ).
Example: inverse(RandomUnimodularMat(ZZ,500))

#15 - 23 Oct 2023 20:41 - John Abbott

I have made inverse interruptible (DetByGauss). I suppose we should also make a DetByCRT at some point?