Quick update on crashes and valgrind

Andrea Valassi (CERN)

Madgraph on GPU development meeting, 25th June 2024

https://indico.cern.ch/event/1355155

(previous update was on June 04 before my holidays – only mentioning changes since then)

SIGFPE crash in rotxxx

- There is a SIGFPE crash #855 in Fortran function rotxxx (aloha_functions.f)
  - Only in optimized –O3 code: relevant variables in gdb show up as <optimized out>
  - Disabling optimization (IIRC –O1 is enough?) makes the crash disappear
  - My proposed workaround #857: add the *volatile* keyword for a few Fortran variables
    - Disable optimizations of very specific lines of code (related to Fortran SIMD?)
    - *This technique is extensively used in cudacpp SIMD ixx/oxx: volatile prevents many crashes*
  - Issue and fix are fully reproducible (crashes without, does not crash with volatile)
  - Not clear why it appears only for some iconfig – but I would fix this independently
    - And fixing this issue then makes it possible to see further issues down the line...

One crash I see

NB: this is not an “intermittent” crash that may sometimes be reproduced and sometimes may not: it is a crash I see all the time...

```
Program received signal SIGFPE, Arithmetic exception.
rotxxx (p=..., q=..., prot=...) at aloha_functions.f:1247
1247              prot(1) = q(1)*q(3)/qq/qt*p1 -q(2)/qt*p(2) +q(1)/qq*p(3)
#0  rotxxx (p=..., q=..., prot=...) at aloha_functions.f:1247
#1  0x00000000004087e0 in gentcms (pa=..., pb=..., t=-181765.47706865534, phi=0.64468537567405615, ma2=0, m1=234.1712866912786,
    m2=210.15563843880372, p1=..., pr=..., jac=3.0327734872026782e+25) at genps.f:1480
#2  0x0000000000409849 in one_tree (itree=..., tstrategy=<optimized out>, iconfig=104, nbranch=4, p=..., m=..., s=..., x=...,
    jac=3.0327734872026782e+25, pswgt=1) at genps.f:1167
#3  0x000000000040bb84 in gen_mom (iconfig=104, mincfig=104, maxcfig=104, invar=10, wgt=0.03125, x=..., p1=...) at genps.f:68
#4  0x000000000040d1aa in x_to_f_arg (ndim=10, iconfig=104, mincfig=104, maxcfig=104, invar=10, wgt=0.03125, x=..., p=...)
    at genps.f:60
#5  0x000000000045c865 in sample_full (ndim=10, ncall=32, itmax=1, itmin=1, dsig=0x438b00 <dsig>, ninvar=10, nconfigs=1,
    vecsize_used=16384) at dsample.f:172
#6  0x000000000043427a in driver () at driver.f:257
#7  0x000000000040371f in main (argc=<optimized out>, argv=<optimized out>) at driver.f:302
#8  0x00007ffff743feb0 in __libc_start_call_main () from /lib64/libc.so.6
#9  0x00007ffff743ff60 in __libc_start_main_impl () from /lib64/libc.so.6
#10 0x0000000000403845 in _start ()
```

- Olivier (thanks!) tried to reproduce it on my machine itscrd90 but was unable to
  - **Can you please try again? I gave step-by-step instructions on github...**
- I understand that adding volatile is controversial: hence I do more tests with valgrind
- Note: I have no evidence at all that this may be related to iconfig-ichannel mapping

Some results from valgrind

- First I tested the pure Fortran madevent_fortran through valgrind: found two issues
  - A minor leak in driver.f (file opened and not closed)
    - Issue https://github.com/mg5amcnlo/mg5amcnlo/issues/109
    - Fix https://github.com/mg5amcnlo/mg5amcnlo/pull/110 - for review OM
  - An uninitialized variable goodjet in reweight.f (possible undefined behaviour)
    - Issue https://github.com/mg5amcnlo/mg5amcnlo/issues/111
    - Workaround https://github.com/mg5amcnlo/mg5amcnlo/pull/112 - for review OM
    - Not a fix! A real fix is needed... the code is accessing a variable that was not properly defined!
  - After adding the patches for these two issues, valgrind is happy on madevent_fortran
    - HOWEVER: madevent_cpp is still crashing in rotxxx, no progress in this respect
- **Now running valgrind on madevent_cpp**
  - This is taking forever (40 minutes and not over yet)... hangs??? is there an infinite loop?
- Note: I also considered address sanitizer as suggested long ago by StephanH
  - On Fortran code, however, I do not get any useful results (from gfortran)
    - We do not support flang yet for Fortran...
  - On C++ code, I need to fix some issues with clang build options
    - Surprise however: with clang and without address sanitizer, the code also seems to hang???

A snapshot of other issues

- There is a different SIGFPE crash #845 in cudacpp function sigmakin
  - Intermittent: same binary executable sometimes crashes and sometimes does not...
  - This seems most likely related to the wrong/missing iconfig-ichannel map for colors?
- There is a color mismatch #856 in LHE files
  - This is clearly related to the wrong/missing iconfig-ichannel map for colors
- There is still the zero cross section I reported in #826
  - Olivier's patch does not fix this for me
- Olivier mentioned a few issues he identified in tests with Stefan
  - Can you provide a detailed reproducer please?
  - Some of these may be related to what I described, some may not
- Last point: IMO it is imperative that we have QUICK systematic tests of "launch"
  - See discussion in #711: allow generate_events with lower precision i.e. fewer events
  - We did not agree on this last year – I think the issues we see now are a consequence of that
  - This would have tested systematically all iconfig for all processes
- My opinion: we need tests, tests, tests...

Another crash I see

NB: this is an "intermittent" crash: it sometimes crashes and sometimes does not...

Open Intermittent FPE "erroneous arithmetic operation" in gqttq tmad test (in random color selection within sigmakin) #845

valassi opened this issue on May 16 · 5 comments

```
Missing separate debuginfos, use: dnf debuginfo-install glibc-2.34-60.el9.x86_64 libgcc-11.3.1-4.3.el9.alma.x86_6
(gdb) where
#0  0x00007ffff7f98d6f in mg5amcCpu::sigmaKin (allmomenta=0x7ffff76bf040, allcouplings=0x7ffff7b57040, allrndhel=
    allrndcol=0x6300d00, allMEs=0x6310d80, channelId=channelId@entry=1, allNumerators=0x6341000, allDenominators=
    allselhel=0x6320e00, allselcol=0x6330e80, nevt=16384) at CPPProcess.cc:1189
#1  0x00007ffff7f9fa3e in mg5amcCpu::MatrixElementKernelHost::computeMatrixElements (this=0x6340ee0, channelId=ch
    at MatrixElementKernels.cc:115
#2  0x00007ffff7fa52d2 in mg5amcCpu::Bridge<double>::cpu_sequence (goodHelOnly=false, selcol=0x7fffffc1cb50, selh
    mes=0x7fffffc3cb50, channelId=1, rndcol=0x7fffffc9ceb0, rndhel=0x7fffffcbceb0, gs=0x1d35a68 <strong_+8>, mome
    this=0x62e0a70) at /usr/include/c++/11/bits/unique_ptr.h:173
#3  fbridgesequence_ (ppbridge=<optimized out>, momenta=<optimized out>, gs=0x1d35a68 <strong_+8>, rndhel=0x7ffff
    rndcol=0x7fffffc9ceb0, pchannelId=<optimized out>, mes=0x7fffffc3cb50, selhel=0x7fffffc2cb50, selcol=0x7fffff
#4  0x0000000000043008c in smatrix1_multi (p_multi=<error reading variable: value requires 2621440 bytes, which is
    hel_rand=<error reading variable: value requires 131072 bytes, which is more than max-value-size>,
    col_rand=<error reading variable: value requires 131072 bytes, which is more than max-value-size>, channel=1,
    out=<error reading variable: value requires 131072 bytes, which is more than max-value-size>, selected_hel=..
    vecsize_used=16384) at auto_dsig1.f:618
#5  0x0000000000431c11 in dsig1_vec (all_pp=<error reading variable: value requires 2621440 bytes, which is more
    all_xbk=<error reading variable: value requires 262144 bytes, which is more than max-value-size>,
    all_q2fact=<error reading variable: value requires 262144 bytes, which is more than max-value-size>,
    all_cm_rap=<error reading variable: value requires 131072 bytes, which is more than max-value-size>,
    all_wgt=<error reading variable: value requires 131072 bytes, which is more than max-value-size>, imode=0,
    all_out=<error reading variable: value requires 131072 bytes, which is more than max-value-size>, vecsize_use
#6  0x0000000000432d48 in dsigproc_vec (all_p=...,
    all_xbk=<error reading variable: value requires 262144 bytes, which is more than max-value-size>,
    all_q2fact=<error reading variable: value requires 262144 bytes, which is more than max-value-size>,
    all_cm_rap=<error reading variable: value requires 131072 bytes, which is more than max-value-size>, iconf=1,
    symconf=..., confsub=..., all_wgt=<error reading variable: value requires 131072 bytes, which is more than ma
    all_out=<error reading variable: value requires 131072 bytes, which is more than max-value-size>, vecsize_use
#7  0x000000000433b1f in dsig_vec (all_p=..., all_wgt=..., all_xbk=..., all_q2fact=..., all_cm_rap=..., iconf=1,
    all_out=..., vecsize_used=16384) at auto_dsig.f:327
#8  0x000000000044a922 in sample_full (ndim=7, ncall=8192, itmax=1, itmin=1, dsig=0x433d10 <dsig>, ninvar=7, ncon
    at dsample.f:208
#9  0x0000000000042ebc0 in driver () at driver.f:256
#10 0x0000000000040371f in main (argc=<optimized out>, argv=<optimized out>) at driver.f:301
#11 0x00007ffff743feb0 in __libc_start_call_main () from /lib64/libc.so.6
#12 0x00007ffff743ff60 in __libc_start_main_impl () from /lib64/libc.so.6
#13 0x0000000000403845 in _start ()
(gdb) l
1184              const int ievt = ievt00 + ieppV;
1185              //printf( "sigmaKin: ievt=%4d rndcol=%f\n", ievt, allrndcol[ievt] );
1186              for( int icolC = 0; icolC < ncolor; icolC++ )
1187              {
1188    #if defined MGONGPU_CPPSIMD
1189                const bool okcol = allrndcol[ievt] < ( targetamp[icolC][ieppV] / targetamp[ncolor - 1][ieppV]
1190    #else
1191                const bool okcol = allrndcol[ievt] < ( targetamp[icolC] / targetamp[ncolor - 1] );
1192    #endif
```

For the moment I assume this is a DIFFERENT ISSUE (in gq_ttq) from the rotxxx crash (in gg_ttgg)...

Note: unlike the rotxxx crash, this is clearly related to iconfig-ichannel mapping

A snapshot of other issues

- There is a different SIGFPE crash #845 in cudacpp function sigmakin
  - Intermittent: same binary executable sometimes crashes and sometimes does not...
  - This seems most likely related to the wrong/missing iconfig-ichannel map for colors?
- There is a color mismatch #856 in LHE files
  - This is clearly related to the wrong/missing iconfig-ichannel map for colors
- There is still the zero cross section I reported in #826
  - Olivier's patch does not fix this for me
- Olivier mentioned a few issues he identified in tests with Stefan
  - Can you provide a detailed reproducer please?
  - Some of these may be related to what I described, some may not
- Last point: IMO it is imperative that we have QUICK systematic tests of "launch"
  - See discussion in #711: allow generate_events with lower precision i.e. fewer events
  - We did not agree on this last year – I think the issues we see now are a consequence of that
  - This would have tested systematically all iconfig for all processes
- My opinion: we need tests, tests, tests...

A crash others see

Can we have a reproducer for this please?
Which process, which commands...
(In GENERAL: can we have reproducers please before committing fixes?...)

Anyway: I assume it is related to my #845 crash and will test this against it...

And maybe it might fix the rotxxx crash too, but I would not bet on it...