ctools - Bug #1258

Model caching is actually not thread save (impact on ctobssim)

07/09/2014 02:57 PM - Knödlseder Jürgen

Status:	Closed	**Start date:**	07/09/2014
Priority:	Immediate	**Due date:**	
Assigned To:	Knödlseder Jürgen	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	3rd coding sprint		

Description

While trying to run simulations on of 4 observations in a container (using cspull), I encountered the following spurious segmentation fault:

```
Thread 0:  Dispatch queue: com.apple.main-thread
0   libSystem.B.dylib             0x00007fff8191039e madvise + 10
1   libSystem.B.dylib             0x00007fff818d97fd large_malloc + 799
2   libSystem.B.dylib             0x00007fff818cb709 szone_malloc_should_clear + 3390
3   libSystem.B.dylib             0x00007fff818ca98a malloc_zone_malloc + 82
4   libSystem.B.dylib             0x00007fff818c8c88 malloc + 44
5   libstdc++.6.dylib             0x00007fff80280f05 operator new(unsigned long) + 97
6   libgamma.0.dylib              0x00000001011b0bb9 std::vector<GCTAEventAtom, std::allocator<GCTAEventAtom> >::reserve(unsigned long) + 121
7   libgamma.0.dylib              0x00000001012089a9 GCTAModelRadialAcceptance::mc(GObservation const&, GRan&) const + 793
8   libctools.0.dylib             0x00000001020229bd ctobssim::simulate_background(GCTAObservation*, GModels const&, GRan&, GLog*) + 461
9   libctools.0.dylib             0x0000000102026ff4 _ZN8ctobssim3runEv.omp_fn.0 + 676
10  libctools.0.dylib             0x0000000102025ea2 ctobssim::run() + 818
11  _ctools.so                    0x0000000101fee056 _wrap_ctobssim_run + 102 (ctools_wrap.cpp:1430)
12  org.python.python             0x00000001000ba468 PyEval_EvalFrameEx + 28696
13  org.python.python             0x00000001000b960a PyEval_EvalFrameEx + 25018
14  org.python.python             0x00000001000bb215 PyEval_EvalCodeEx + 2197
15  org.python.python             0x00000001000b91ad PyEval_EvalFrameEx + 23901
16  org.python.python             0x00000001000b960a PyEval_EvalFrameEx + 25018
17  org.python.python             0x00000001000b960a PyEval_EvalFrameEx + 25018
18  org.python.python             0x00000001000b960a PyEval_EvalFrameEx + 25018
19  org.python.python             0x00000001000bb215 PyEval_EvalCodeEx + 2197
20  org.python.python             0x00000001000bb336 PyEval_EvalCode + 54
21  org.python.python             0x00000001000e018e PyRun_FileExFlags + 174
22  org.python.python             0x00000001000e0449 PyRun_SimpleFileExFlags + 489
23  org.python.python             0x00000001000eff9d Py_Main + 2909
24  org.python.python             0x0000000100001f14 0x100000000 + 7956

Thread 1:
0   libSystem.B.dylib             0x00007fff818cc1e0 OSSpinLockUnlock + 0
1   libgamma.0.dylib              0x000000010102f2ca GVector::free_members() + 26
2   libgamma.0.dylib              0x00000001010a6852 GSkyDir::rotate_deg(double const&, double const&) + 434
3   libgamma.0.dylib              0x00000001011fa0f8 GCTAModelRadialGauss::mc(GCTAInstDir const&, GRan&) const + 264
4   libgamma.0.dylib              0x00000001012089f4 GCTAModelRadialAcceptance::mc(GObservation const&, GRan&) const + 868
5   libctools.0.dylib             0x00000001020229bd ctobssim::simulate_background(GCTAObservation*, GModels const&, GRan&, GLog*) + 461
6   libctools.0.dylib             0x0000000102026ff4 _ZN8ctobssim3runEv.omp_fn.0 + 676
7   libctools.0.dylib             0x000000010200fac6 gomp_thread_start + 230
8   libSystem.B.dylib             0x00007fff818fffd6 _pthread_start + 331
9   libSystem.B.dylib             0x00007fff818ffe89 thread_start + 13

Thread 2:
0   libSystem.B.dylib             0x00007fff818f50ff sin + 399
1   libgamma.0.dylib              0x00000001011fa08a GCTAModelRadialGauss::mc(GCTAInstDir const&, GRan&) const + 154
2   libgamma.0.dylib              0x00000001012089f4 GCTAModelRadialAcceptance::mc(GObservation const&, GRan&) const + 868
3   libctools.0.dylib             0x00000001020229bd ctobssim::simulate_background(GCTAObservation*, GModels const&, GRan&,
 GLog*) + 461
4   libctools.0.dylib               0x0000000102026ff4 _ZN8ctobssim3runEv.omp_fn.0 + 676
5   libctools.0.dylib               0x000000010200fac6 gomp_thread_start + 230
6   libSystem.B.dylib                 0x00007fff818fffd6 _pthread_start + 331
7   libSystem.B.dylib                 0x00007fff818ffe89 thread_start + 13

Thread 3 Crashed:
0   libSystem.B.dylib                 0x00007fffffe007f7 __memcpy + 87
1   libgamma.0.dylib                  0x0000000101002179 std::vector<double, std::allocator<double>
 >::_M_insert_aux(__gnu_cxx::__normal_iterator<double*, std::vector<double, std::allocator<double> > >, double const&) + 265
2   libgamma.0.dylib                  0x000000010113fb83 GModelSpectralFunc::mc_update(GEnergy const&, GEnergy const&) const +
 2115
3   libgamma.0.dylib                  0x00000001011417f3 GModelSpectralFunc::mc(GEnergy const&, GEnergy const&, GTime const&,
 GRan&) const + 275
4   libgamma.0.dylib                  0x0000000101208a9a GCTAModelRadialAcceptance::mc(GObservation const&, GRan&) const +
 1034
5   libctools.0.dylib               0x00000001020229bd ctobssim::simulate_background(GCTAObservation*, GModels const&, GRan&,
 GLog*) + 461
6   libctools.0.dylib               0x0000000102026ff4 _ZN8ctobssim3runEv.omp_fn.0 + 676
7   libctools.0.dylib               0x000000010200fac6 gomp_thread_start + 230
8   libSystem.B.dylib                 0x00007fff818fffd6 _pthread_start + 331
9   libSystem.B.dylib                 0x00007fff818ffe89 thread_start + 13
```

Here, 4 threads were created, and the 4th crashed during a memory copy. Repeating the experiment led to different threads crashing, but always at the same memcpy instruction.

I finally recognized that models are part of GObservations, hence the model instances were shared by the threads. In other words, the threads accessed concurrently the models. This leads to problems for cashed values, as threads try to update the cash concurrently.

In GObservations::likelihood::eval the problem is prevented by making copies of the model, but this is not the case for ctobssim.

Making a copy of the models container within the #pragma omp parallel section solves the problem.

History

#1 - 07/09/2014 02:57 PM - Knödlseder Jürgen

- Project changed from GammaLib to ctools

- Target version deleted (3rd coding sprint)

#2 - 07/10/2014 11:25 PM - Knödlseder Jürgen

- Target version set to 3rd coding sprint

#3 - 07/20/2014 11:16 PM - Knödlseder Jürgen

- Status changed from Resolved to Closed