Profiling memory allocations

Giulio Eulisse

Northeaster University, Boston (MA), U.S.A.

There are three kinds of algorithms:

- those which are slow
- those which take a large amount of memory
- most of them though achieve to do both

Besides raw speed we are interested in profiling memory allocations which in some cases is even more important than computational cost.

If you don't have CPU time, you can wait, if you don't have RAM you have to buy it.

We are interested in profiling:

- unintrusively, we don't want to modify our target code
- with low overhead, both computationally and from memory footpring point of view

MemProfLib, as found in IGUANA $>$4.2.3 is a little tool, four-hand-coded with Lassi Tuura, which uses glibc features to keep track of all the memory allocation performed by a program.

As requested:

- Low overhead (we profiled both IGUANA and OSCAR).
- Not very memory hungry.

glibc comes handy to do this kind of profiling. It provides a way to hook into memory related primitives:

- `__malloc_hook`
- `__realloc_hook`
- `__free_hook`
- `__memalign_hook`

They are called on the call of associated C fuctions.

- The library implements the malloc hooks.
- It is loaded before everything else via `LD_PRELOAD` mechanism.
- Whenever a `malloc()` call occurs, it gets the point in which it happened (via `backtrace()`), and stores all the information in a tree structure.
- The tree is dumped to a gprof style text file.

- `__malloc_hook`s are glibc only.
- However most `dlsym()` implementations support `RTLD_NEXT` option, which allows for generic wrapping of system calls. Using it in place of `__malloc_hook` would result in FreeBSD, Solaris portability.
- Similar tricks can be done on Windows. See (MSJ Feb 2002 ”Under the hood”).

Since oprofile does not provide call-tree structure, we decided to add a JProf like profiling option to the profiler.

- It uses `SIG_ALARM` (`SIG_PROF`, actually) to do the sampling.
- It uses `LD_PRELOAD`.
- Same kind of output.

- Multi-thread support (we need to wrap `clone()`).
- Make it glibc independent.

gprof style output:

```
-------------------------------------------------------
    492651028 465775672 1/1  G4SteppingManager::InvokePSDIP(...) ( libG4tracking.so ) [631]
[0] 465775672 465750904 1   G4Transportation::PostStepDoIt(...) (libG4transportation.so)
         1032      1032 1/1  G4Allocator< ... >::MallocSingle() (libG4tracking.so) [1912]
        23736     23736 1/1  G4Navigator::LocateGlobalPointAndSetup(...) (libG4volumes.so) [1269]
-------------------------------------------------------
```

- JPROF:

  http://www.mozilla.org/performance/jprof.html

- MemProfLib temporary web-page:

  http://eulisse.web.cern.ch/eulisse/oprofile/