My thesis' main ideas

Juan Francisco Martnez — juan.martinez@bsc.es

August 31, 2017

This is just a draft document devoted to gather the general MIRI thesis ideas to develop during the Q4. The idea of my master thesis cames to me through Judit. She has detected a need in the performance analysis methodology when the structure of the analyzed application is important in order to report a feedback to the customer and ease the process of who to blame in code. In most cases the access to the code is not allowed to the analyzer so we have to think in a manner to infer the application code through its behaviour.

The way to infer the application internal structure is by mean of a post-mortem paraver trace analysis. The paraver tracefile has information about the used program model in addition to some hardware counter information. With the program model info also can be found information about the callstack and the code line from where every level of the stack has been called. In order to have this last information available you must to compile your code with debug information.[1]

Note that the debug information requirement is like a contradiction because if you have debug information you can reconstruct the code withour wierd behavioral analysis (with DWARF or STABS libraries instead). The end of this approach is just because if we have the callstack information with function names and files we can compare the inferred internal structure with the actual one. Once the methodology has been validated, the binaries to study will be without debug information so another way to identify univocally every call will be proposed.

This approach is just the first out of two steps. The second and most interesting is to be able to infer the intern structure of an application not with information of the whole execution but just with several monitors scattered over the actual execution. I'm not sure if I will reach this point, but in principle is in the roadmap.

1 Thesis' questions

According to this website[2] one of the firsts steps is to consider a set of questions interesting to be answered by this thesis and for the scientific community. Make

[1] e.g. `-g` flag in `GNU C Compiler`.

[2] `http://www.wikihow.com/Write-a-Master\%27s-Thesis`

sure that the questions are not already answered because with the thesis we want to increase the knowledge in a genuine way. Even though, some of this questions can be answered during the state of the art analysis. In this case we can try to propose many new ones.

- Internal structure of an application can be infered just from its behaviour?
- The same behaviour can be explained by different intern structures? So aliasing could appear?
- If yes, with how many frequency?
- This thesis is HPC oriented. With this bounded subset of applications, we can consider a sort of idiosyncracy?
- This idiosyncracy can allow us to perform some assumptions in order to ease the intern structure detection process?
- How the compiler optimizations (-Ox) perturb the intern structure analysis?
- Internal structure of an application can be inferred without the debug information?
- A signal with a repetitive pattern can be inferred just with scattered monitors?
- How to end up with a readable pseudocode without the debug information?
- The internal structure detection can help to performance analyzers with behaviour and code correlation?
- This is an advantage at report time? Why?