Exjobb: Automatically Finding and Explaning Bugs (in collaboration with Axis Communications)

Wanted: Two students for an M.Sc. project. You should have taken one of the Compilers courses, be familiar with C and (ideally) C++, and be curious about the LLVM compiler infrastructure and static program analysis.

Finding bugs earlier saves developers time and headaches. One of the main techniques for finding bugs is *static program analysis*, which can detect bugs by analysing the code structure, i.e., without ever running the code. In collaboration with Axis Communications, we have built a prototype tool that can detect incorrect uses of the `glib` library, which is a fundamental library used by the Gnome and GIMP Open Source projects, as well as by GStreamer, a central piece of Open Source software used by Axis.

Merely knowing that there is a bug in a certain line of a certain code module is not enough to help developers fix the bug, though. Bugs that we can report during verification with tools such as ours are essentially disagreements between two specifications — but this disagreement may not be obvious. Consider the following code:

```
1  int div(T v) {
2      return 10 / v.f; // division by zero found here!
3  }
4
5  c.f = 1;
6  div(c); // no bug
7  a.f = 0;
8  b = a;
9  div(b); // bug!
```

A typical static bug checker will report a potential division by zero in line 2. This isn't ideal: First, not all calls to `div` will trigger the error, so we should connect the warning to line 9 and make clear that the code in line 6 is safe. Second, for the programmer to understand the cause of the error, they need to understand how the `0` from line 7 flows *indirectly* into variable `b` in line 8.

In this project, you will:

- Familiarise yourself with the Phasar program analysis framework and with our existing analysis tool
- Extend our existing static analysis tool to track the *provenance* of bug reports, i.e., which observations and which assumptions led to the bug report.
- Evaluate the cost of your provenance tracking by examining its runtime overhead
- Evaluate the effectiveness of both your provenance tracking and the underlying program analysis through case studies

This project is a collaboration between Axis Communications and the Software Development and Environments group.

Supervisors at Axis: David Svensson Fors (davidsf@axis.com) and Karin Hedlund
Supervisors at LTH: Christoph Reichenbach (christoph.reichenbach@cs.lth.se), and co-supervisor Alexandru Dura (alexandru.dura@cs.lth.se)

Contact us for details!