Lab

Abstract Interpretation, Lab and Project

1.1 Exercises with Pagai

`http://pagai.forge.imag.fr/`

Preparation On the ENS machines, we have installed PAGAI, the binary is in the directory:

`/soft/enseignants/Laure.Gonnord/pagai/pagai_git/src/pagai`

Copy and untar the archive:

`/soft/enseignants/Laure.Gonnord/CR10/tpai.tgz`

Test pagai with `gopan.c` example. first compiled with `clang` (On the ENS machine, clang 3.4 is available in `/usr/bin`.)

```
clang -emit-llvm -g -c gopan.c -o gopan.bc
</path/to/>pagai -i gopan.bc
```

or simply : `</path/to/>pagai -i gopan.c`.

Finding invariants Play with Pagai (and the various abstract domains) to find numerical invariants :

- For the examples in the tar file. Do you manage to find a suitable invariant for the gaz burner example ? for the diamond example ?
- For hand-written examples that show the need for the various abstract domains. (intervals, octogons, polyhedra).

1.2 Project

In the tgz we provide you a parser and AST construction for a mini-java language; as well as an example of use of the Apron Library. You have to write an analyser for this mini language, that uses Apron to compute operations on the abstract values, and which performs abstract interpretation on the AST of the program:

- forward propagation, using intervals or polyhedra (choice on the command line)
- standard Apron widenings, applied at all loops heads, from the begining or delayed (choice on the command line)
- stdout printing of the invariants in a readable way, for instance only on loops heads. (and log in a .log file, choice of verbosity on the command line).

This code will come with experiments on hand-written and well chosen test files, and a study of at least:

- The scalability of your code.
- The precision of the different abstract domains provided by Apron.

This experimental evaluation will be explained in your report.

Compiling and running at ENSL In your shell:

```
OPAMDIR=/soft/enseignants/Laure.Gonnord/opam
export OPAMROOT=$OPAMDIR/root
PATH=$OPAMDIR/bin:$PATH
eval $(opam config env)
```

On your own machines Type:

```
opam install apron
```

and follow the instruction

Link The best way to compile with Apron is then e.g.

```
ocamlfind ocamlopt -package apron -package apron.polkaMPQ -c analyzer.ml
```

The best way to link with Apron is then e.g.

```
ocamlfind ocamlopt -linkpkg -package apron -package apron.polkaMPQ bidules.cmx -o analyzer
```

Additional (optional) features

- Deal with local variables: use these local variables only in their declaration scopes. First, only compute invariants for each function independently.
- Add a way to deal with function calls (interprocedural analysis). (This is not trivial).
- Add a way to deal with arrays.

Modalities This is an individual work due on **october, xx 2016**. Send an email to **Laure.Gonnord** containing a tar file (.tgz) with:

- your (commented) code, that should compile on the ENS machines.
- a README explaining how to run your analysis.
- a bench of tests.
- a report in pdf (3 to 6 pages). Explain algorithms, implementation choices, the experimental study.