

A (very) Short Introduction to Analyses and Techniques

Manuel Oriol

Axiomatic semantics

- To prove pieces of code.
- Infer properties on the code.
- Example able to prove (possibly with a prover) the values of postconditions coming from the precondition.

$$\begin{array}{l} \{x \neq 0\} \\ x := x + 1 \\ x := x / 2 \\ \{x \neq 0\} \end{array}$$

Software Metrics

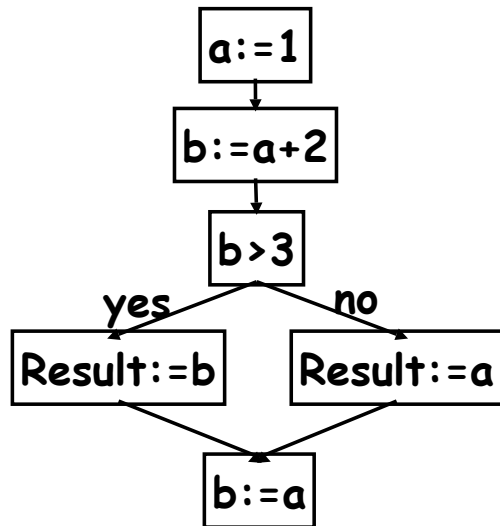
- Number of methods
- Length of Mehtods
- Dependencies
- Number of Lines of Code
- Cyclomatic Complexity
- ...

Testing Components

- Coverages (path, code, data...)
- Model-based (check that results are correct according to a model)
- Random
- Mutation Testing

Dataflow Analysis

As we could see in previous examples:



Control-Flow Graph

- There is a need for an analysis of data manipulations
- Traditionally, it leads to optimizations (e.g. statements that have no influence may be removed), loops may have statements extracted from them etc...
- Based on all paths through programs
- It is the analysis of labels by excellence

Program slicing

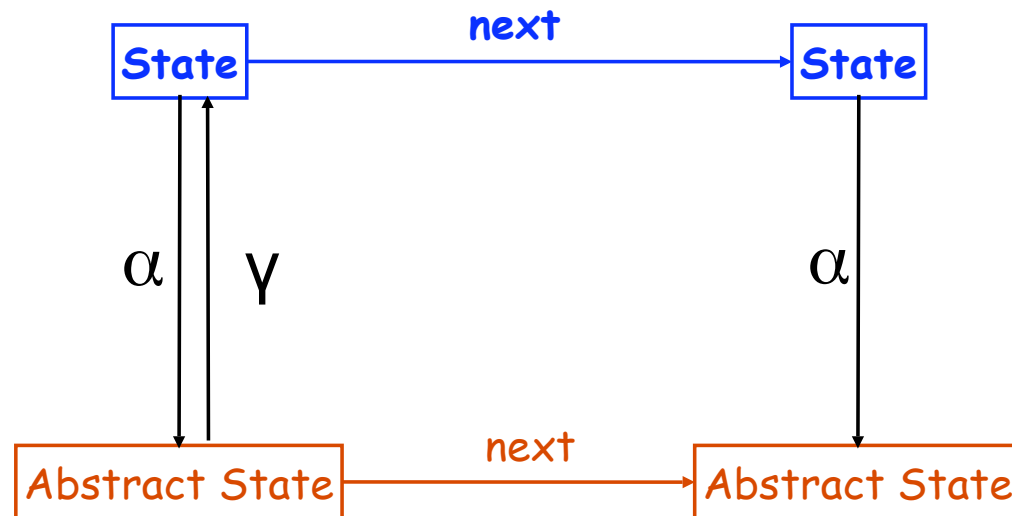
What are the statements leading to the value of b at the end?

```
a:=1
b:=5
if (b>3) then
  Result:=b
else
  a:=2
end
b:=a
```

Abstract Interpretation

A technique for analyzing the programs by modeling their values and operations.

In fact it is an execution that one can make to prove facts.



Model Checking

- Let's try all the inputs!
- well... there are so many inputs that we have to restrain... by using a model
- E.g. use a boolean abstraction and verify for all boolean values

Conclusions

- Lots of techniques coding a subset of one is already a significant task