Modelling Program Verification Tools for Software Engineers

Sophie Lathouwers and Vadim Zaytsev
s.a.m.lathouwers@utwente.nl, vadim@grammarware.net

Motivation
There are many program verification tools and techniques but it is difficult to find and choose a suitable tool.

Techniques are hard to understand
Tools are often not production-ready
Information sources may be unreliable and unavailable

Contribution
We present a megamodel [1] for program verification tools to make them more accessible to a broader audience. We have also prepared a data set with concrete examples of program verification tools.

Megamodel
The model is based on the classic division of roles in a correctness proof as introduced by Goldwasser, Micali and Rackoff in 1985 [2].

Using this model, we have identified seven levels: PV0 – PV6. The higher levels tend to give more correctness guarantees, though typically at the cost of more user effort. For example, PV6 includes theorem provers, whereas PV1 may include a tool to check emptiness of automata.

Data set
Choose data sources
Identify tools in papers
Collect data
Define megamodel
Classify tools
Identify trends

- 460 papers from CAV and TACAS
- 420+ tools, frameworks and specification formats
- Includes information such as domain, what input is needed, input format, output produced, internal details, relations to other tools, links to project pages, and more!

Available at https://slebok.github.io/proverb