17-665 Program Analysis
Project
Evelyn Kuo

Project Proposal

I will be working alone on an empirical evaluation of several open-source tools targeting Python with respect to automated test generation of Python programs. I chose Python as the programming language of choice due to its popularity and difficulty as a dynamically typed language. The general outline for evaluation would include a qualitative measuring of ease of use (learning curve, readability of outputs), amount of extra code needed to prepare code for automatic test generation, amount of extra code needed to generate automatic tests, quantitatively how good it is at evaluating the correctness on a shared set of programs, and qualitatively what other benefits the tool may provide. Tools which I am currently considering evaluating include pynguin, deal, TSTL, CrossHair, Klara, and Auger which support automatic test generation with varying degrees of setup work. I think evaluating automatic test generation in Python is interesting because Python is a very popular language and writing good tests is an integral yet often overlooked part of writing good programs. For the checkpoint, I plan to try out the various tools on simple examples / provided tutorials in order to understand at a basic level how each of the tools work and make sure my environment is set up to run the tools. Several links to quickstarts are provide below

- https://pynguin.readthedocs.io/en/latest/user/quickstart.html
- https://deal.readthedocs.io/details/examples.html
- https://github.com/agroce/tstl
- https://crosshair.readthedocs.io/en/latest/introduction.html
- https://klara-py.readthedocs.io/en/latest/quick_start.html
- http://chrislaffra.blogspot.com/2016/12/auger-automatic-unit-test-generation.html

17-665 Program Analysis
Project
Evelyn Kuo

Project type, title, members in group
Outline what going to do, why interesting, how evaluate
checkpoint
Python Tools https://github.com/analysis-tools-dev/static-analysis#python
Need to check if still in use

pyflakes
Type checker mypy
Contract https://deal.readthedocs.io/
Code refactoring https://pybowler.io/
Prospector https://prospector.landscape.io/en/master/
Python in other languages - quick setup to make for all languages?

- wenyan lang
    - https://spectrum.ieee.org/classical-chinese
    - https://wy-lang.org/
- Chinese python
    - http://reganmian.net/blog/2008/11/21/chinese-python-translating-a-programming-language/
    - https://github.com/gasolin/zhpy
        - twpy
        - Adding in static analysis tools onto it?
- https://en.wikipedia.org/wiki/Non-English-based_programming_languages

Carry out an empirical evaluation of one or more existing analysis method(s) and/or opensource tool(s) (typically 3–5 such tools).

For the proposal, what type of analysis will you implement/compare, for what language, and using what framework(s)? How will you evaluate success (at a high level, i.e., what's your general test strategy), and/or what metrics or approach will you be using to compare frameworks, methods, or tools? You may be weighing different options, still; if so, be concrete about what those options are and how you will choose between them. E.g., if you want to target C, you may want to choose between CIL or Clang as a framework (or others...). You need not decide in the proposal, but you do need to at least list what options you are considering. For the checkpoint, concretize any decisions you did not make in the proposal, provide evidence of progress in the implementation direction (e.g., evidence that you have compiled a toy analysis in the framework if it is new to you, or that you have set up baseline or simple testing of analysis or gotten various comparative tools to compile or run on a simple test case).