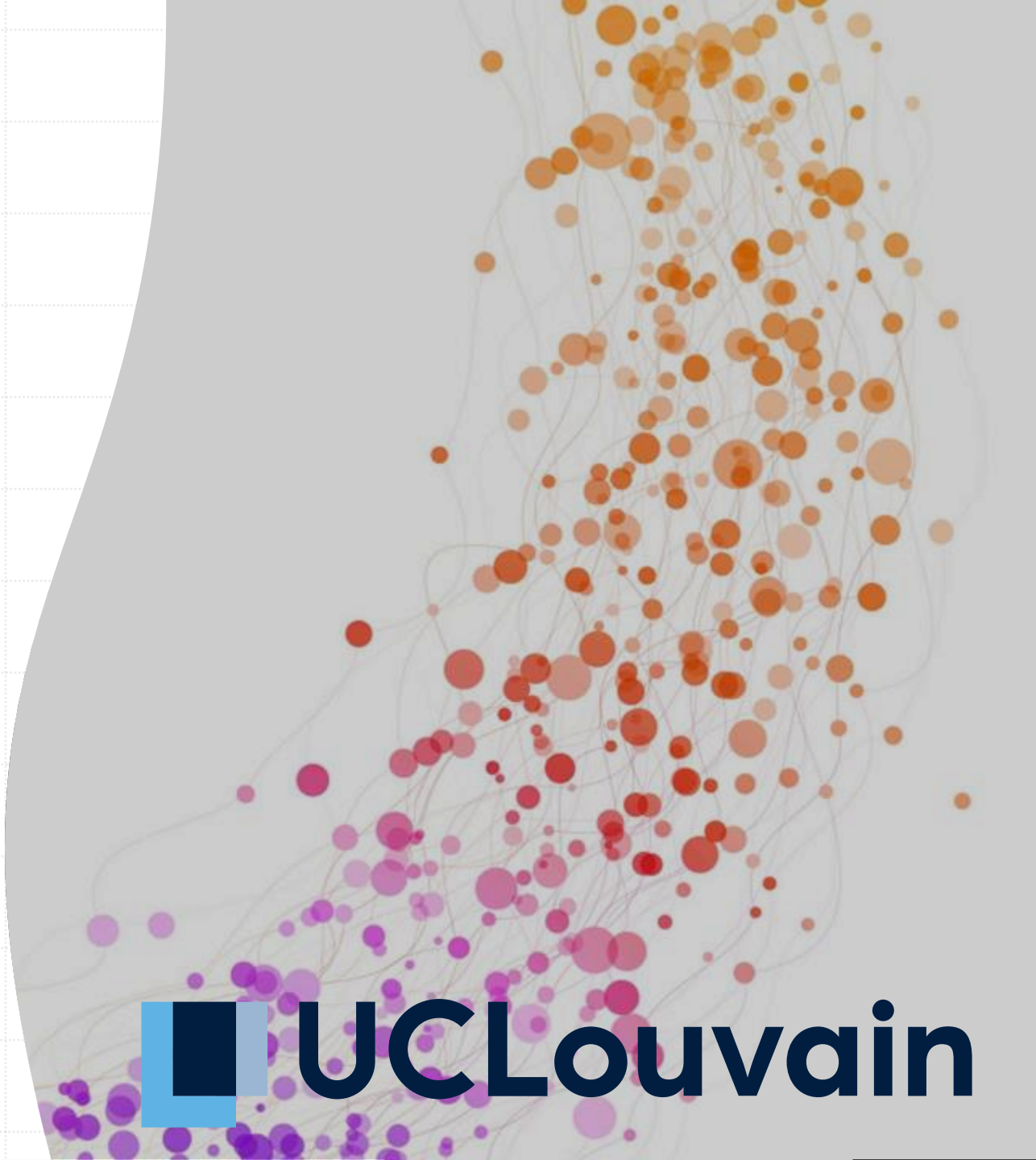
Pyttern: a Python-Based Program Query Language

Julien Liénard, kim Mens,
Siegfried Nijssen



UCLouvain

Context

Student make mistakes

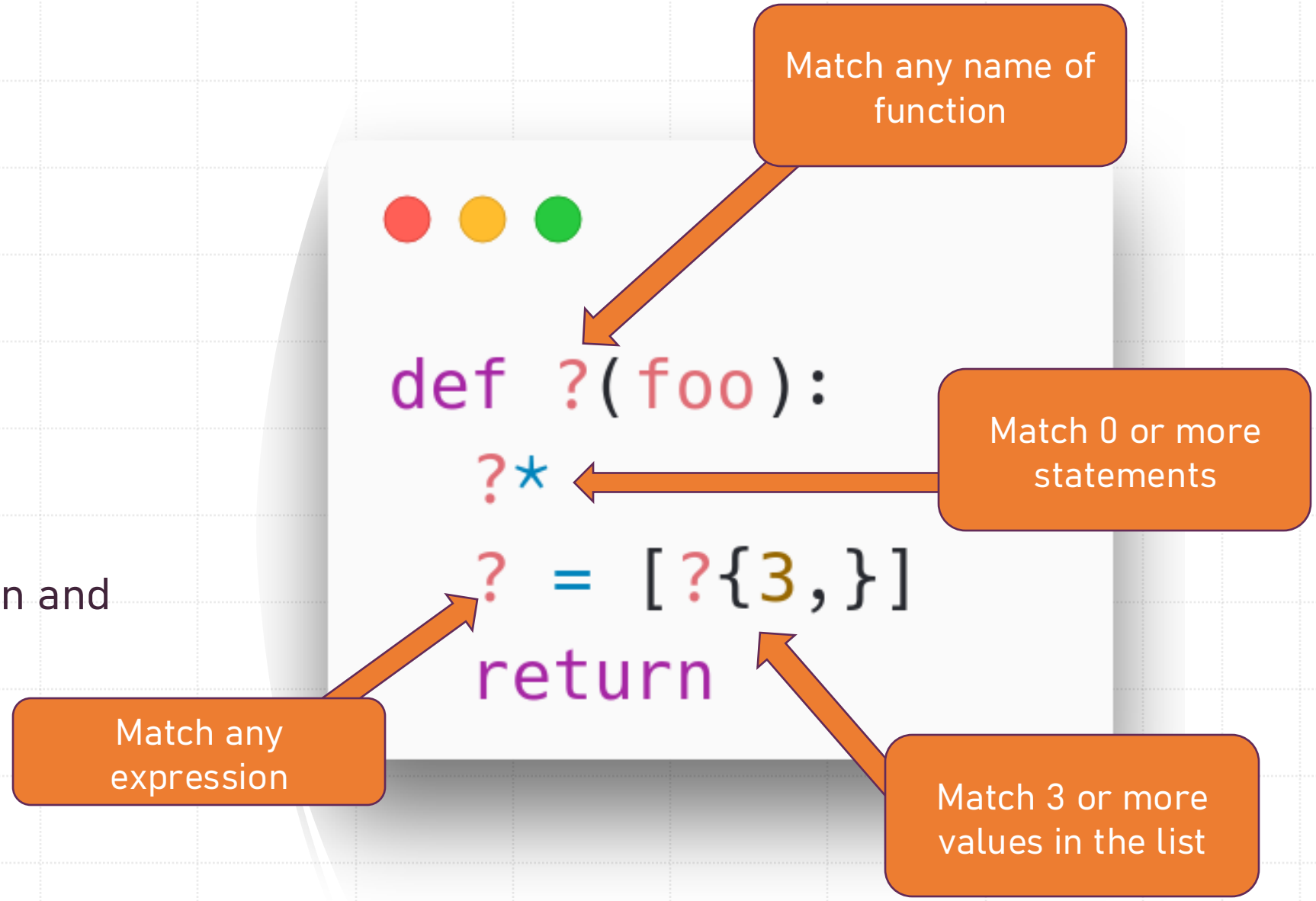
Student have misconception and bad smells in their code

How to detect them efficiently?

Lots of PQL but can be hard to learn

What is Pyttern?

- Language to describe patterns
- Build over Python
- Build to be easy to learn and use
- Inspired by regex



Some examples

Match any number of arguments

```
def facteurs(?*):
    ?:*
        for ? in range(?*):
            if ? == 0:
                ?*
```

Match the body at any level of identation

Match only While or For expressions

```
def ?(?*):
    ?*
    ?[While, For]:*
        ?*
```

```
def approx_pi(n):
    ?pi = 0
    ?:*
        ?pi = ?pi + ?
    return ?pi
```

Bind the match to the variable "pi"

First version of pyttern

Match AST to AST

Syntax inspired by SCRUPLE

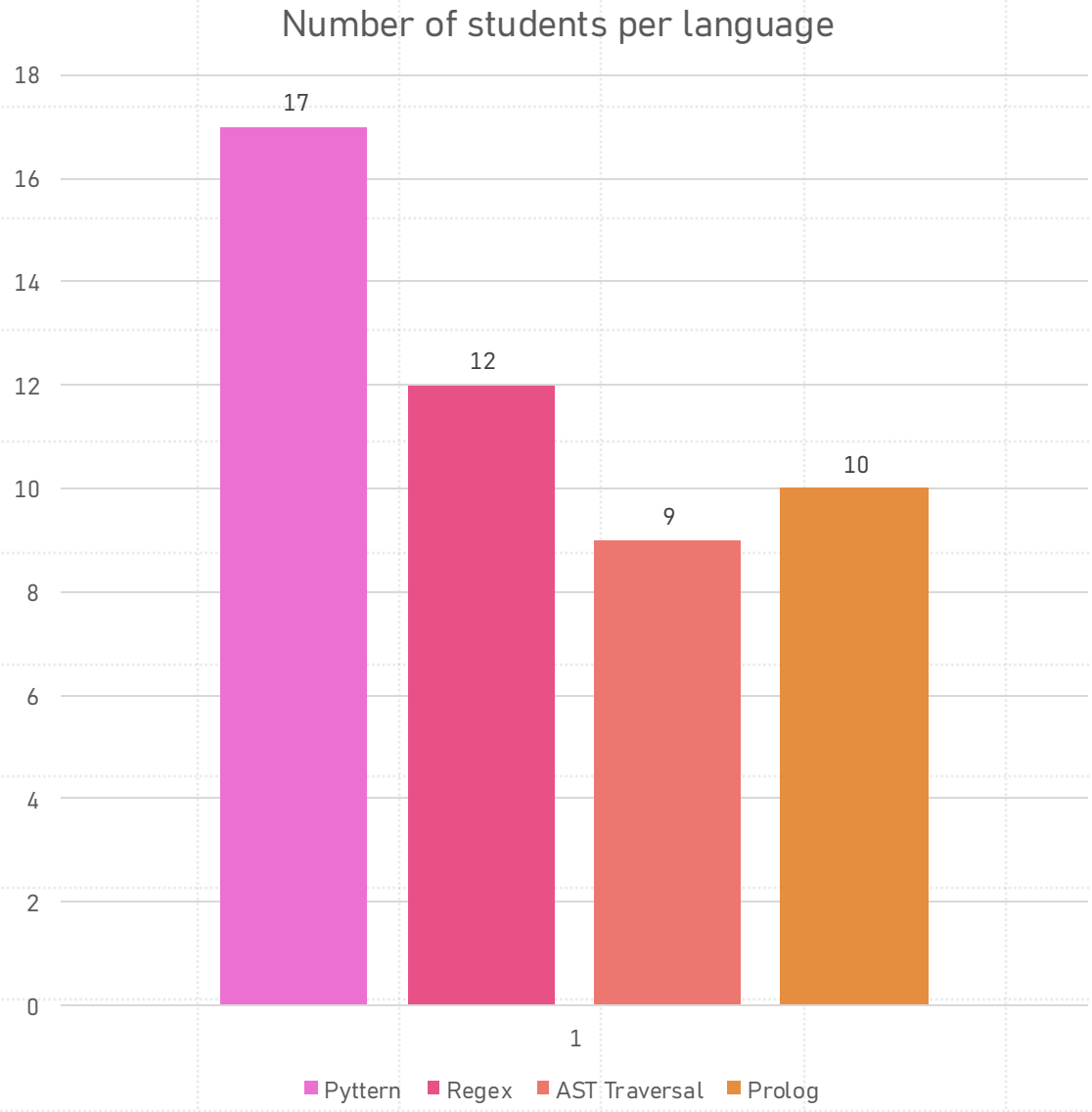
Can be slow

No way of counting the number of match

No debuging/visual tools

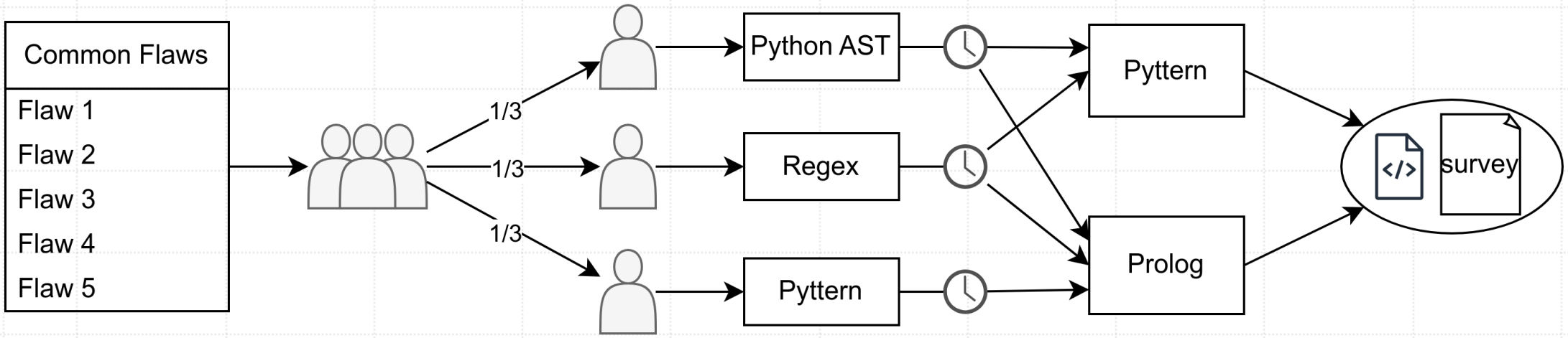
Validation

- First version of Pyttern
- 30 students
- Compare 2 out of 4 techniques:
  - Pyttern
  - Regex
  - AST traversal
  - Prolog
- Anonymous survey



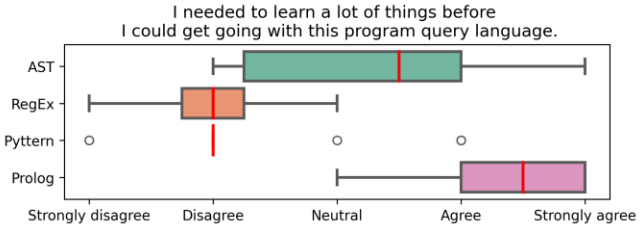
Validation

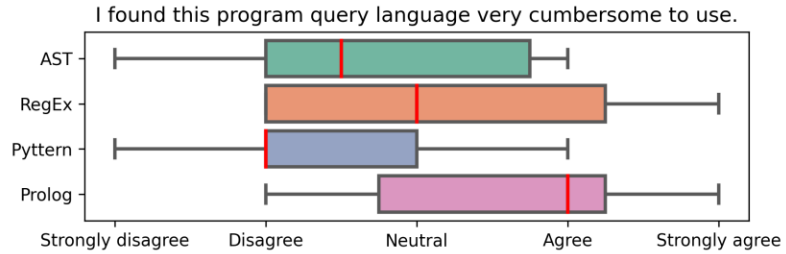
- 2000 generated code with flaws
- Generated from student code

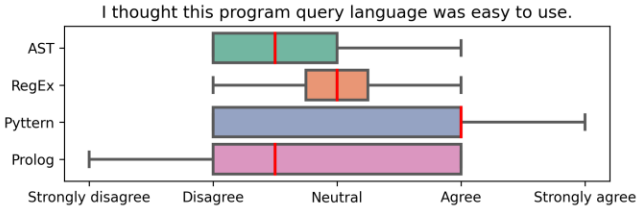


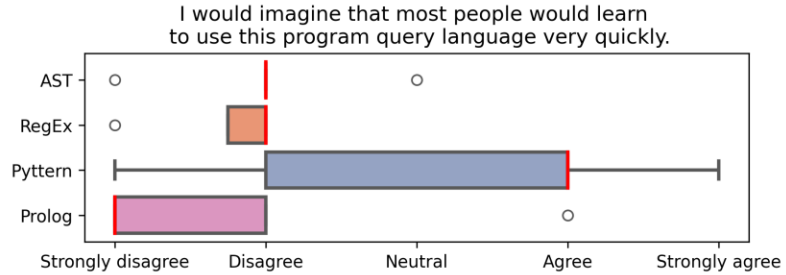
Validation

- Compared to other languages, students found Pyttern:
  - Easy to use
  - Quick to learn



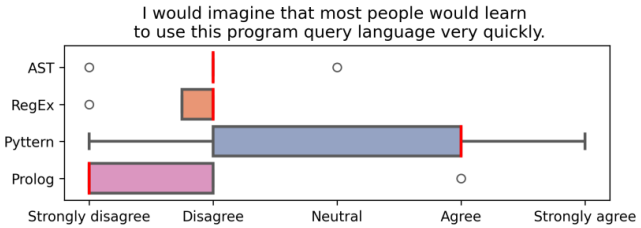


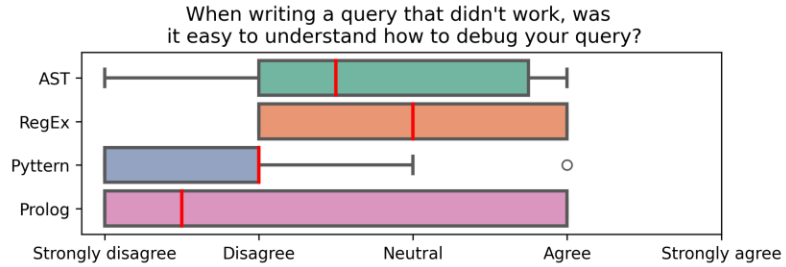


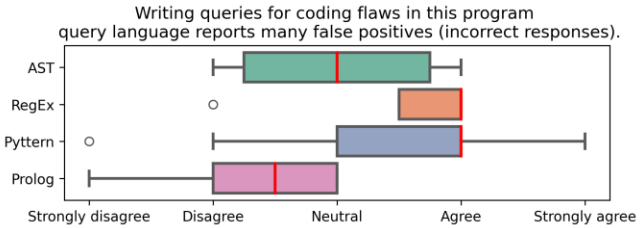


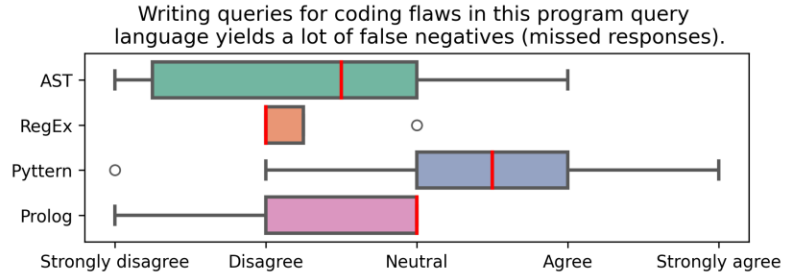
Validation

- But:
  - Hard to debug
  - Not precise enough









Threats to validity

Limited sample

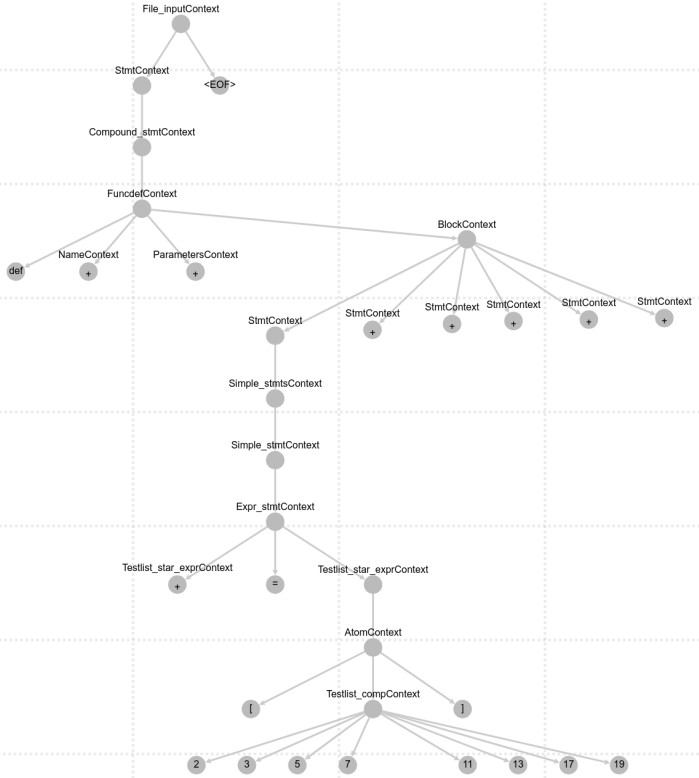
Master students

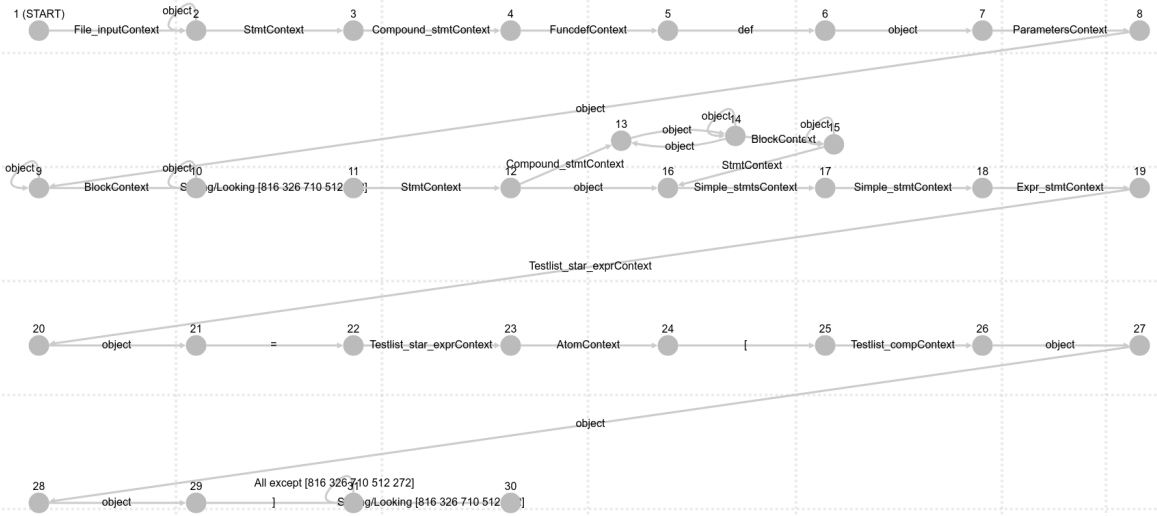
Small bugs in prototype

Ambiguities in pattern instructions

Second version of Pyttern

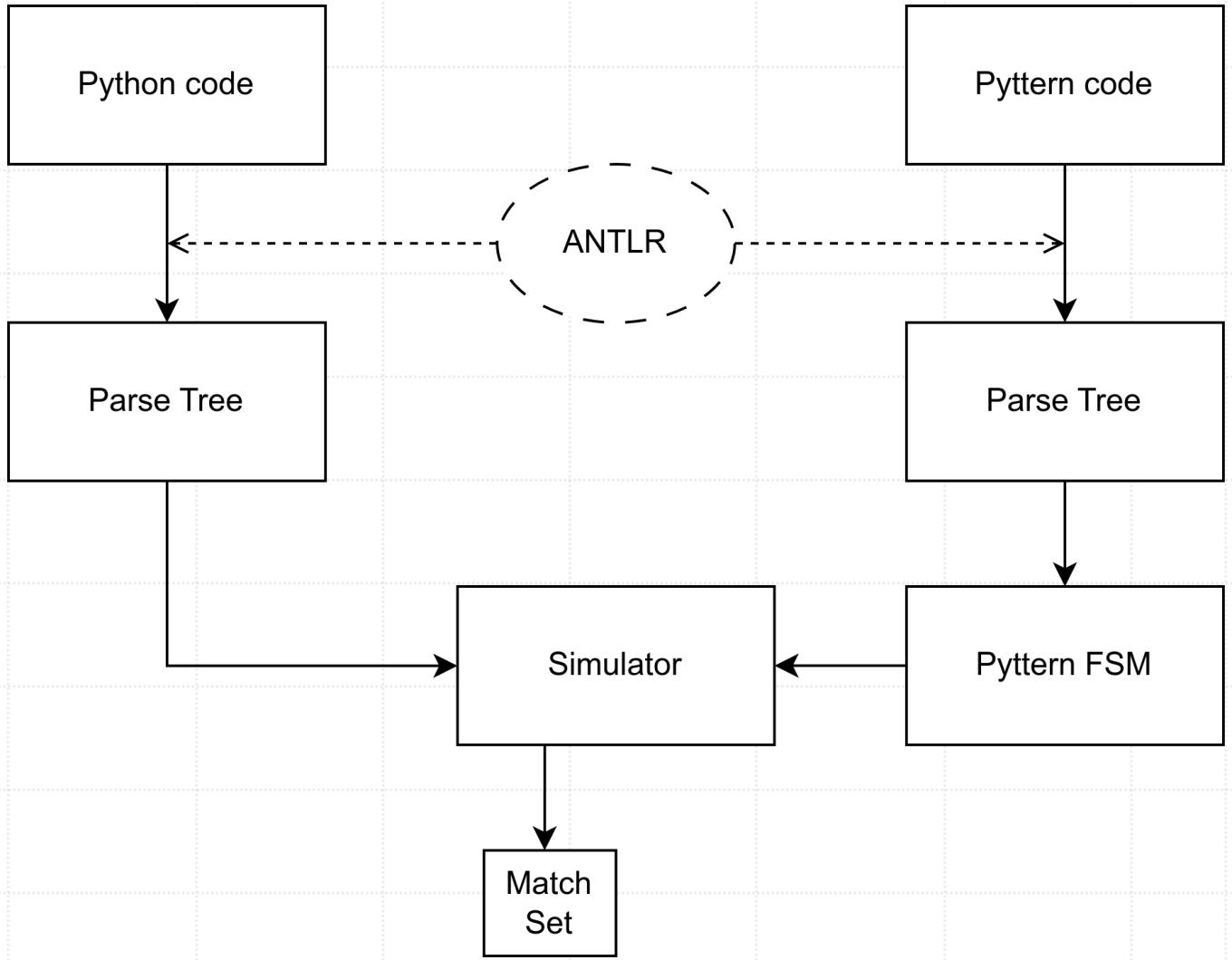
- Architecture inspired by SCRUPLE:
  - Pattern transformed to Finite State Machine (FSM)
  - Code transformed to Parse Tree (PT)





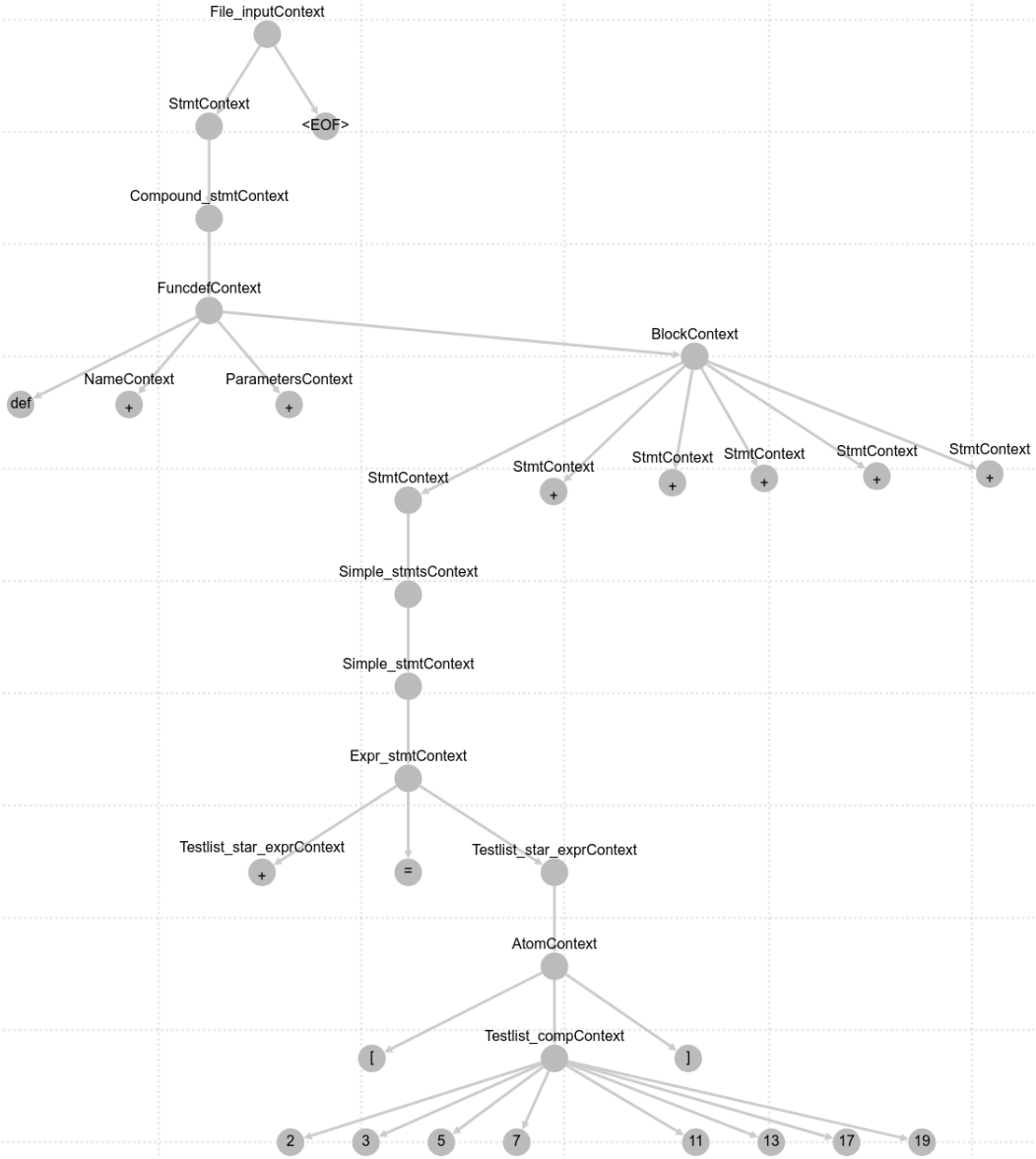
Paul, S. (1992, November). SCRUPLE: A reengineer's tool for source code search. In *Proceedings of the 1992 conference of the Centre for Advanced Studies on Collaborative research-Volume 1* (pp. 329-346).

Code Structure



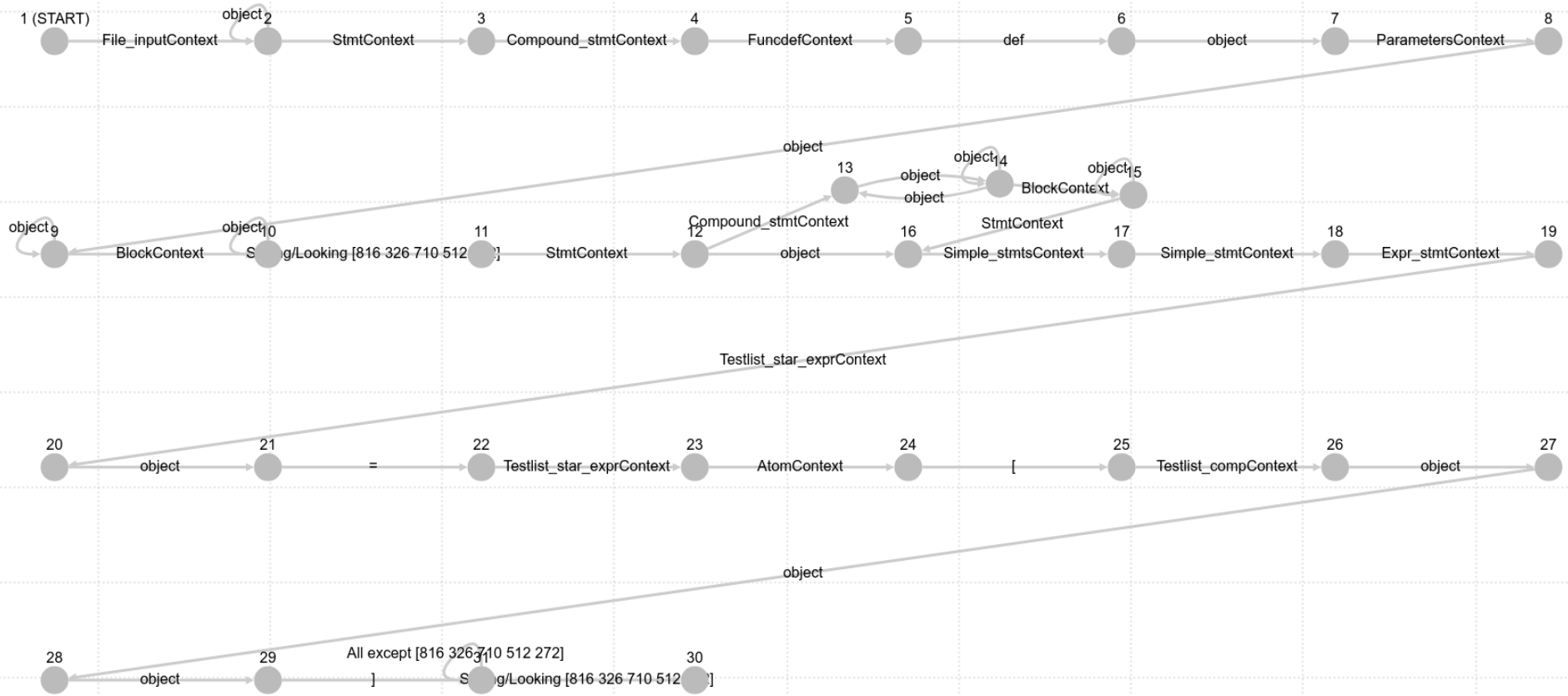
ANTLR Parse Tree

- LL(*) parser
- Easy to use
- Base Python grammar already defined
- Easy to find grammar for other languages



Pyttern FSM

- Based on SCRUPLE
- Non-deterministic FSM
- Some constraint to ensure termination



Matching Algorithm

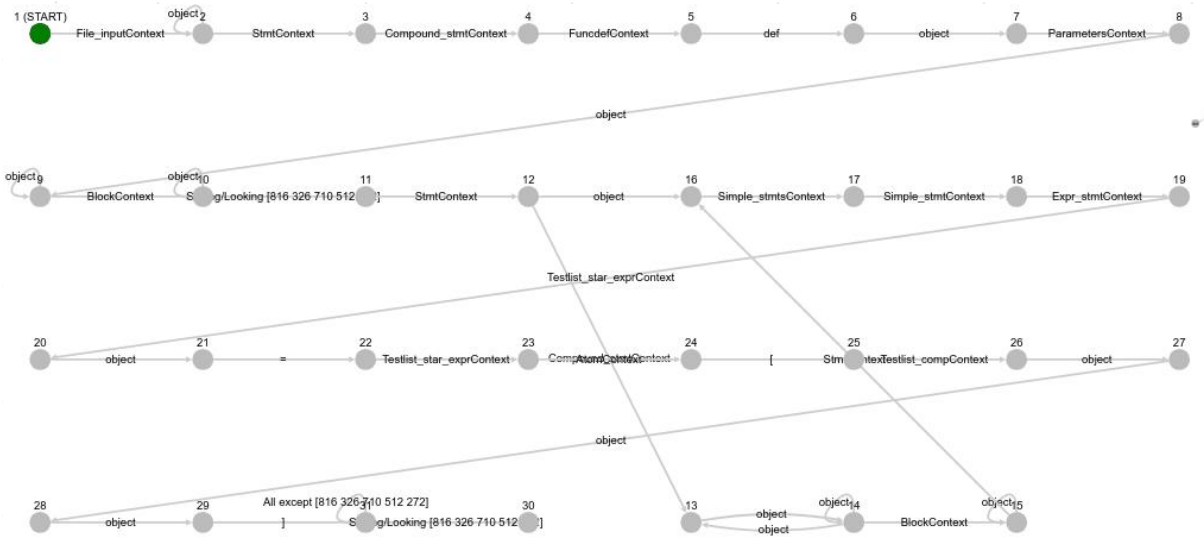
FSM direct the matching algorithm

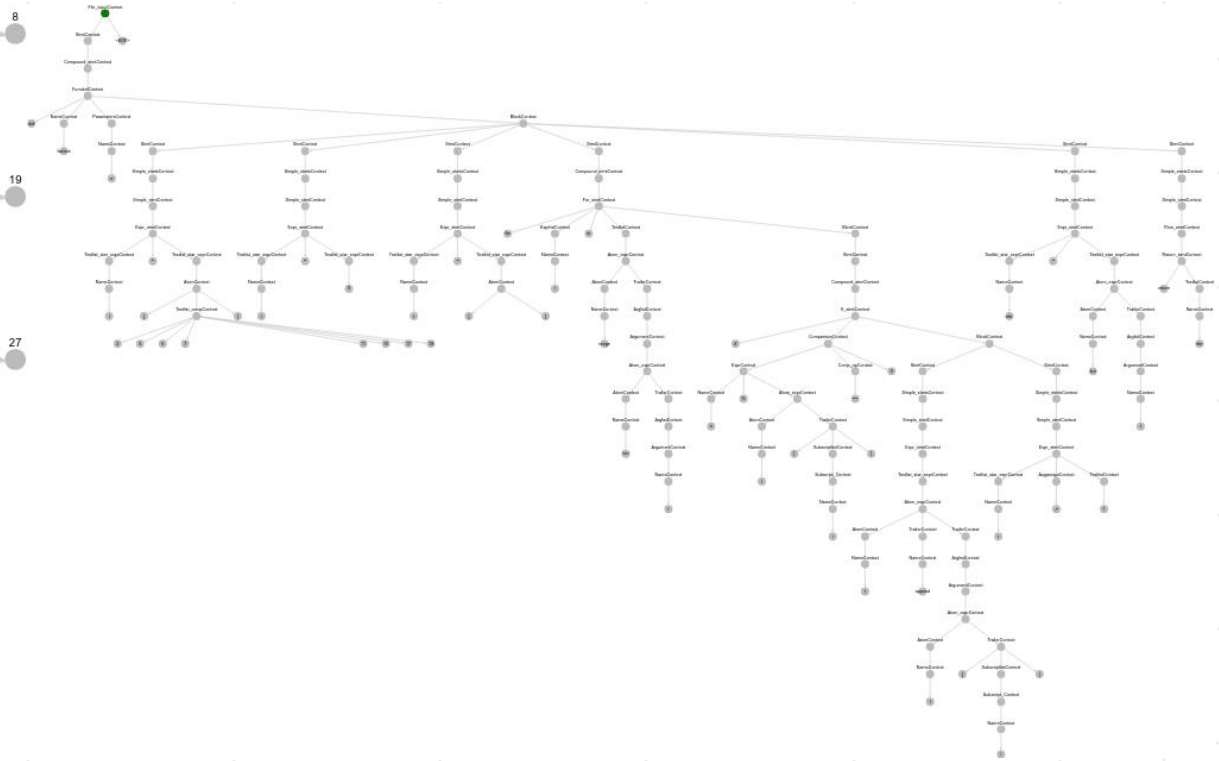
3 Movements possible:

Move to left child	Move to right sibling	Move to parent + move to right sibling

Matching Algorithm

Matching Algorithm

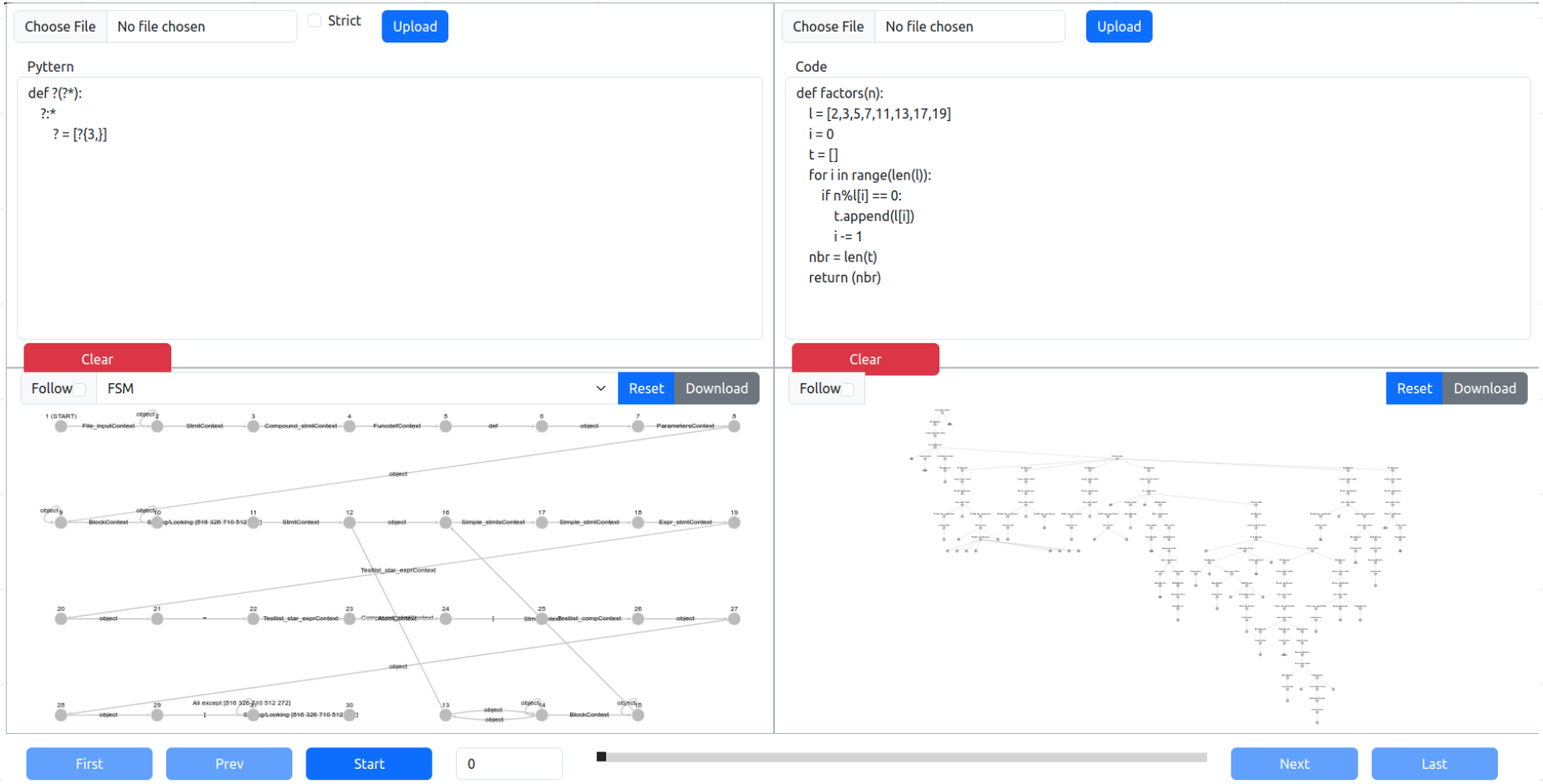




Difference between V1 and V2

- Whole new code structure
- Better matching algorithm
- Debugging/Visual tools
- Designed to be more maintainable
- Designed as a framework for other languages

Difference between V1 and V2



Next steps

New experiment with more student

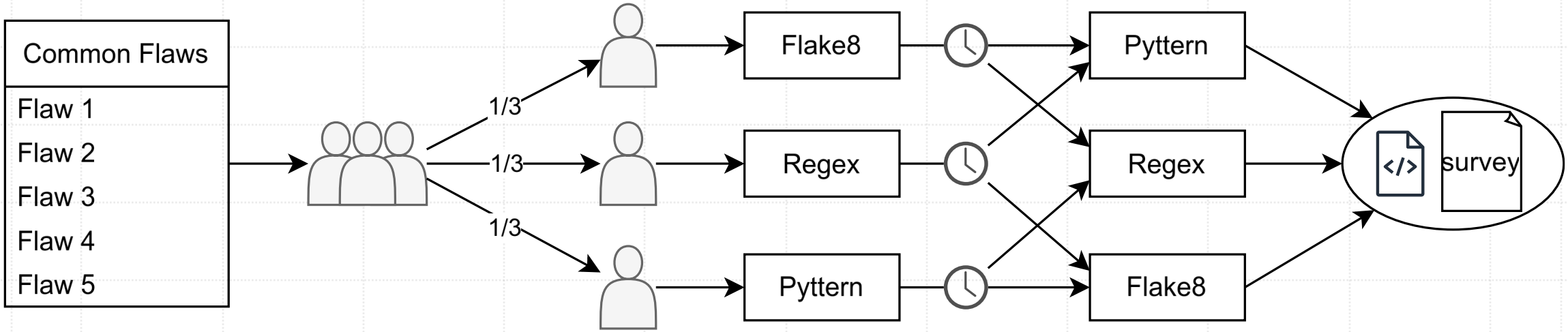
New version of Pyttern

Development of Jattern (Master thesis)

Comparison to other tools like Flake8 or CodeQL (Master thesis)

Validation

- Correct ambiguities
- More students
- New version of Pyttern



Thank you for listening