LapsPython

Extend LAPS to synthesize Python/R

Christopher Brückner & Enisa Sabo

27.06.2022

Backlog

Issues from Sprint 2:

- Rewrite translation module because it got hard to work with ✔
  - Code is beautiful now
- Fix bugs to translate the more complex data we got ✖
  - New bugs introduced
- Problem: Some synthesized programs are hard to understand
  - Nested invented primitives are still the biggest issue
  - Common: Calling functions with wrong (number of) arguments

Example: Nested invented primitive

- LAPS uses “de Bruijn” notation, not LISP notation
    - More compact, but more difficult to understand

- The following program contains (at least) 3 variables **$n**
- **#** marks invented primitive

```
#(λ (λ (λ (#(λ (λ (_rflatten (map $0 (r_split _rdot
$1)))))) $2 (λ (_rconcat $1 $2)))))))
```

Correctness of Translations

- Synthesized programs are saved if they solve all task examples
- Translations are run for the same examples

Correctness of Translations

- Synthesized programs are saved if they solve all task examples
- Translations are run for the same examples

	Old Data
Programs	75
Correct Translations	67 (89%)
Tasks	18
Solved Tasks	17 (94%)

Correctness of Translations

- Synthesized programs are saved if they solve all task examples
- Translations are run for the same examples

	Old Data	New Data
Programs	75	1646
Correct Translations	67 (89%)	84 (5%)
Tasks	18	346
Solved Tasks	17 (94%)	27 (8%)

Correctness of Translations

- Additional metric:
  - Complexity (e.g. number of tokens)

- Solve the simpler programs first
  - Increases probability that complex programs will be solved

```
(λ (_rflatten (_rappend _w (_rsplit _d $0))))
```

```
(λ (#(λ (λ (_rflatten ($1 (_rsplit _rdot $0))))) (λ (#(λ (λ (if (_rmatch (car
(_rrevcdr $0)) $1)))) _e $0 $0 (#(λ (λ (if (_rmatch (car (_rrevcdr $0)) $1)))) _i $0
$0 (#(λ (λ (λ (cons $0 (cons $1 (cdr $2)))))) $0 _n _f)))) $0))
```

Side Note: Correctness of Translations

```
(λ (_rflatten (_rappend _w (_rsplit _d $0))))


def if_the_word_ends_with_any_letter_add_w_after_that(arg1):
    _rsplit_1 = __regex_split('.', arg1)
    _rappend_1 = _rsplit_1 + ['w']
    return "".join(_rappend_1)
```

- Is the Python program correct?

Side Note: Correctness of Translations

```
(λ (_rflatten (_rappend _w (_rsplit _d $0))))
```

```
def if_the_word_ends_with_any_letter_add_w_after_that(arg1):
    _rsplit_1 = __regex_split('.', arg1)
    _rappend_1 = _rsplit_1 + ['w']
    return "".join(_rappend_1)
```

- Is the Python program correct?
    - Yes and no: The translation is correct, the source program is wrong
    - Task only contains example with words ending on letters
    - Dataset is flawed

Summary

- Unfortunately, the main parts of the program still need work
- But: 5 of 9 issues in Sprint 3 are closed
- Leftover:
  - Fixing the translation of string processing (Backlog Sprint 2)
  - Language annotations & translations for list processing