Syntax-Guided Program Synthesize

Problem

- Given grammar ***G*** and constraints ***C***
- Find Program ***P*** s.t. $P \in L(G) \wedge P \sim C$

- Input: String : Synth-lib
- Output: String: SMT-lib

Grammar

- Context-free
- Node with type

```
(synth-fun max2 ((x Int) (y Int)) Int
    ((Start Int (x
                 y
                 0
                 1
                 (+ Start Start)
                 (- Start Start)
                 (ite StartBool Start Start)))
     (StartBool Bool ((and StartBool StartBool)
                      (or  StartBool StartBool)
                      (not StartBool)
                      (<=  Start Start)
                      (=   Start Start)
                      (>=  Start Start)))))
```

Constraints

- SMTlib
- Z3 solvable

```
(declare-var x Int)
(declare-var y Int)

(constraint (>= (max2 x y) x))
(constraint >= (max2 x y) y)
(constraint(or (= x (max2 x y))
               (= y (max2 x y))))

(check-synth)
```

Output

- SMTlib

```
(define-fun max2 ((x Int) (y Int)) Int (ite (<= x y) y x))
```

Provided API

- Easy python(2)
  - Want python3? Do it yourself
  - Include”3.5” in your name(see test.py)
- Not at all robust
  - Debugging to the last sec

Parser & checker

- Parser: SMTlib to Python list

```
benchmarkFile = open(sys.argv[1])
    bm = stripComments(benchmarkFile)
    bmExpr = sexp.sexp.parseString(bm,
parseAll=True).asList()[0]
    pprint.pprint(bmExpr)
```

- Checker:
  - return none if unsat
  - return counter example if sat(type: Z3py model)
  - take string as input

```
if(checker.check(translator.toString(SynFunResult)) == None):
            Ans = Curr
```

Baseline

- Naïve top-down bfs
  - No counter-example used
  - No constraint-solving used(except for verification)
  - Just string parsing
- Max2 solved in 3secs
- 3/33 tests solved in 10secs
- Provided: stupid bfs
  - 1 test solved(three.sl)
  - 2lines different from baseline

Input-Output format

- $yourname(mhzeng)$/main.py
- Must be executable by commandline: python main.py $testfilename(max2.sl)$
  - Argument 1 will be test input file
- Output your answer(in string) to standard output

- Test script provided.

Problem Limits

- Integer\Boolean
- Basic arithmetic: + - * / % == > < ite
- Basic logic: and or not imply
- All tests are open!

Possible solutions

- Top-down search
  - Priority search
- Constraint solving
  - CEGIS
- Learning
  - Probabilistic grammar
- …

Useful tool:Z3py

- Tutorial: http://ericpony.github.io/z3py-tutorial/guide-examples.htm