COMP 597: Assignment #2

Instructor: Xujie Si, TA: Breandan Considine

Due: Nov. 10th, 23:59 in Montreal

This is the 2nd assignment of COMP 597: Automated Reasoning with ML.

1 Implement your own FlashFill (45 points)

Develop your own FlashFill, which supports string and integer arithmetic.

```
      E ::= Var | Const | E + E | E - E | E * E | (E) | if B then E else E
      B ::= E < E | E == E | not B | B and B | B or B | (B)
  Const ::= IntConst | StrConst | "StrLit"
IntConst ::= 0 | 1 | 2 | ... | 9 | IntConstIntConst
 StrLit ::= a | ... | z | _ | IntConst | StrLitStrLit
```

You may design your own internal syntax if you wish, however the synthesized program (i.e., output by your synthesizer) should follow the syntax described above. The semantics of your DSL should resemble the semantics of Python expressions. Special attention is requried for handling multiplication between a string and an integer, as well as addition of two strings, both of which are permitted in Python. An example input is provided below:

```
{
  "Task-1": [
    { "Input": [2, 2], "Output": 4 },
    { "Input": [3, 5], "Output": 15 }
  ],
  "Task-2": [
    { "Input": ["a", "b", "b"], "Output": "aaabb" },
    { "Input": ["aa", "d", "b"], "Output": "aaaaaadb" }
  ],
  "Task-3": [
    { "Input": ["hello", 4, 2], "Output": "hellohello" },
    { "Input": ["world", 1, 0], "Output": "world" },
    { "Input": ["hi", 3, 3], "Output": "" }
  ],
  "Task-4": [
    { "Input": ["plus", 2, 3], "Output": 5 },
    { "Input": ["minus", 5, 3], "Output": 2 },
    { "Input": ["hi", 2, 3], "Output": 0 }
  ]
}
```

For full credit, please provide your source code and instructions to run it.

2 Adapt an existing synthesizer (35 points)

Can you solve (or partially solve) the problems in the above grammar using an existing program synthesizer? For example, you may use a simple enumerative solver [1] or more sophisticated baseline of your choice. A dataset of test cases for string-based program synthesis [2] may be reused from previous years of the SyGuS [3] competition. Please show all work to receive full credit.

3 Compare and contrast (20 points)

Please compare your synthesizer with prior work, discuss the pros and cons (qualatitively) and use concrete examples and performance measurements (quantatively). Your written report should strive to well-written, empirically sound and demontrate a clear understanding of the program synthesis algorithms you have choosen to implement and compare against. The written report should be about 2-3 pages in length with tables and pseudocode.

[1] `https://bitbucket.org/abhishekudupa/eusolver/src/master/`

[2] `https://github.com/SyGuS-Org/benchmarks/tree/master/comp/2017/PBE_Strings_Track`

[3] `https://sygus.org/artifacts/`