

COMP 597: Assignment #2

Instructor: Xujie Si, TA: Breandan Considine

Due: Nov. 3rd

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) | If B then E else E
B ::= E < E | E == E | E > E | NOT B
Const ::= IntConst | StrConst
IntConst ::= 0 | 1 | 2 |
Str ::= a | ... | z | IntConst | StrStr
StrConst ::= "a" | "hello" |
```

Should you like, you may design your own internal syntax, but the synthesized program (i.e., the output of your synthesizer) should follow the syntax described above. The semantics of your DSL should resemble the semantics of Python expressions. Special attention is required for handling multiplication between a string and integer and addition on strings, which are both allowed in Python. An example of the input specification 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": "aaaaadb" }
  ],
  "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 above problems using an existing program synthesizer? For example, you may use a simple enumerative solver ¹ or more sophisticated baseline. 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 (qualitatively) and use concrete examples and performance measurements (quantitatively). Your written report should strive to be well-written, empirically sound and demonstrate a clear understanding of the program synthesis algorithms you have chosen 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/>

²https://github.com/SyGuS-0rg/benchmarks/tree/master/comp/2017/PBE_Strings_Track

³<https://sygus.org/artifacts/>