Exhaustive Symbolic Execution for Common Loops

Solal Pirelli

DSLab

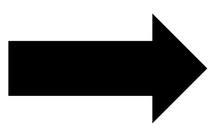
Summarizing loops with common shapes enables exhaustive symbolic execution of more real-world code

Current symbolic execution techniques unroll loops,
they can find bugs but not exhaustively check correctness

Loops can be arbitrarily complex in theory,
but mostly follow common shapes in practice!

Abstracting data structures as maps enables efficient first-order logic

```
for item in lst:
    if item == 42:
        x += 1
    if item < 0:
        break
```



```
assume item in lst
lst_ = lst[0:lst.index(item)]
assume all(x >= 0 for x in lst_)
x += len(list(x for x in lst_ if x == 42))
```

Start with an iteration-based loop, including `break`, `continue`, `return`

Summarize into loop-free and recursion-free code, using first-order predicates and assumptions

Exhaustively symbolically execute the result

Can enumerate all paths
in common data structures

Fully automated,
no need for manual proofs

Want to work on something related? Talk to us!