

TODO Add a name

Kevin Chen

February 12, 2026

1 Introduction

2 Symbolic Execution

Symbolic execution is a way of executing a program abstractly, such that on abstract execution covers multiple possible inputs of the program that share a particular execution path through the code [1]. Each input of symbolically executed code is represented as a symbol rather than assigning it a concrete value. When the value is referenced such as for an if statement, the execution path branches, where one path the symbolic value is evaluated as true while the other path the symbolic value is evaluated as false.

One of the key uses for symbolic execution is in testing. Most unit tests, require the programmer to hardcode in values to test, resulting in various unit tests to cover all branches the code can traverse. On the other hand, a symbolic execution would execute simultaneously on a family of inputs. Consider the following example:

Algorithm 1 Add and Check Even

Require: $x, y \in \mathbb{Z}$

```
1:  $a \leftarrow x + y$ 
2: if  $a \bmod 2 = 0$  then
3:   return error “ $a$  is even”
4: else
5:   return  $a$ 
6: end if
```

3 Conclusion

References

- [1] Jonathan Aldrich. Symbolic execution. Lecture notes 14, Course 17-355, Carnegie Mellon University, 2018. Accessed: 2025.