Jan 17th Report

By Xiang Chen (Echo)

Jan 17, 2020

Contents

1 Type Between Integer and Real Number

For the type correctness of integer and real number, modify the type checker as follows:

- Type correct:
  - $i : REAL = 2.1$
  - $i : REAL = 2$
  - $j : INTEGER = 2$
- Not type correct:
  - $i : INTEGER = 2.1$

Which means when the variable is declared as REAL type, it could be assigned the INTEGER value and REAL value.

However, if the variable is declared as INTEGER type, it cannot be assigned the REAL value.

---

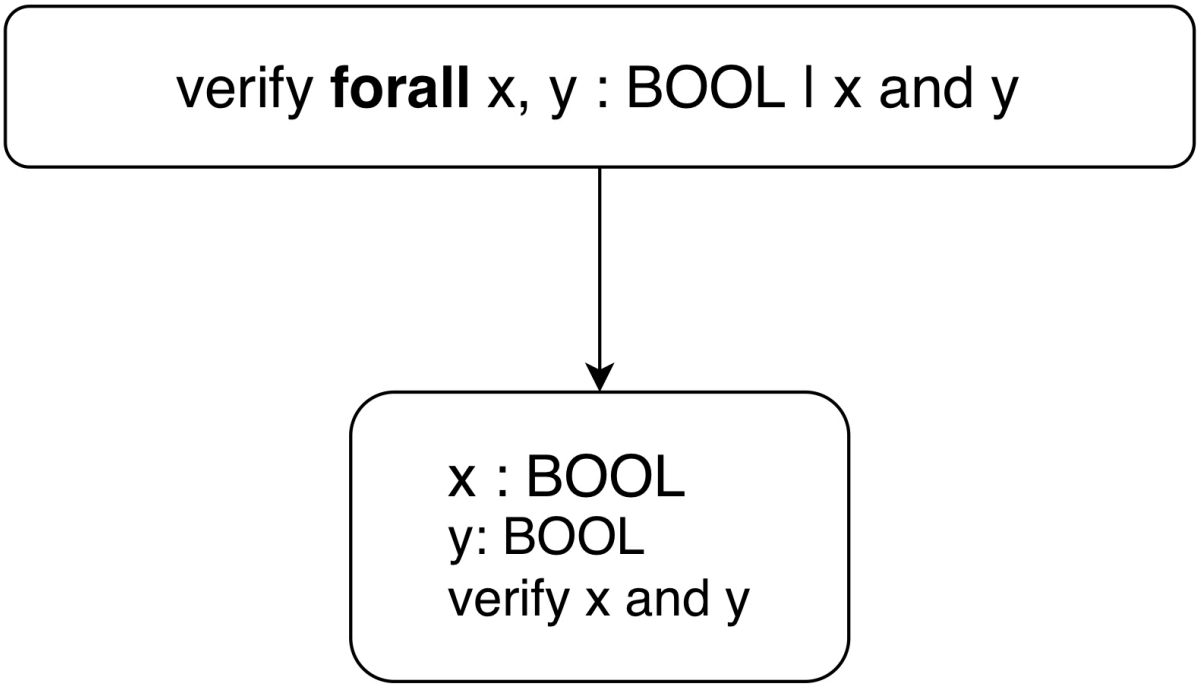
2 Quantification: Forall

```
1 (assert (not (forall((x Int))(=> (< x 0) (< x -1)))))
2 (check-sat)
3 (get-model)
4
```

'►' shortcut: Alt+B

```
sat
(model
)
```

For the quantification, bacause Z3 cannot provide the correct counterexample, I find a way to transform **forall** into normal propositional logic as follows:



For the formula that contains forall, simply grap all the variables in that quantification, and declare them as global variables, and verify the formula directly.

---

3 Array

3.1 Array Declaration and Verification Format

- Declaration: **a : ARRAY[INTEGER]**
- Verification: **a[i] = j**

3.2 Array Index

Z3 supports array index as real numbers, but in my program, I modify my TypeChecker class to make sure the index is of INTEGER type, and start from 1.

For example, **a[i * 2]** is supported, then for Arrays, I need to type check its index first to make sure the index is of INTEGER type.