

Demo of AlgoVer

Institut für Theoretische Informatik

sumAndMax/Bounds.1

sumAndMax/Null.1

sumAndMax/Bounds.2

sumAndMax/Bounds.3

sumAndMax/Null.2

sumAndMax/InitInv

sumAndMax/InitInv.1

sumAndMax/InitInv.2

sumAndMax/loop/else/Inv

sumAndMax/loop/else/Inv.1

sumAndMax/loop/else/Inv.2

sumAndMax/loop/else/Dec

sumAndMax/loop/else/Bounds

sumAndMax/loop/else/Bounds.1

sumAndMax/loop/else/Null

sumAndMax/loop/then/Inv

sumAndMax/loop/then/Inv.1

sumAndMax/loop/then/Inv.2

sumAndMax/loop/then/Dec

sumAndMax/loop/then/Bounds

! Edit

✓ Edit

✓ Edit

✓ Edit

✓ Edit

✓ Edit

! Edit

! Edit

✓ Edit

✓ Edit

✓ Edit

✓ Edit

✓ Edit

✓ Edit

✓ Edit

✓ Edit

✓ Edit

✓ Edit

✓ Edit

✓ Edit

```

7  sum := a[0];
8  max := a[1];
9
10 var i: int := 1;
11 while {i < a.Length}
12   invariant 0 <= i && i <= a.Length
13   invariant forall k: int :: 0 <= k && k < i ==> a[k] <= max
14   invariant i * max >= sum
15   decreases a.Length - i
16 {
17   if {a[i] > max}
18   {
19     max := a[i];
20   }
21   sum := sum + a[i];
22   i := i + 1;
23 }
24 }
25

```

- Proofs conducted on different levels
 - the code itself (needs insight if not closing)
 - the logic level (needs relation back to original program)
- state of the art user interfaces focus only on one level, supporting features for other levels often insufficient
- may hinder users from using/understanding it

- allows to inspect different parts of the proof in individual views
- supports insight into unfinished proof attempts
- seamless transition between different views
- conduction proofs possible via
 - direct manipulation
 - script based
 - annotation based

Demo

DEMO

- more rules and strategies
- better smt-support
- feedback back to source code
- evaluation on larger examples
- ...