Lecture 9 - February 3

ProgTest1 *Guide*, Math Review

Implementation Correctness
Completeness of Contracts:
Pre-condition vs. Post-condition

Announcements/Reminders

- ProgTest1 guide released
- Mockup Test scheduled during lab on Thursday, Feb. 6
- Lab1 solution released
- Lab2 released
- Office Hours: 3pm to 4pm, Mon/Tue/Wed/Thu
- TA contact information (on-demand for labs) on eClass

Correct Algorithm and Complete Postcondition (1.1)

```
------- MODULE ExampleModule -------
EXTENDS Integers, Sequences, TLC
CONSTANT input
-- algorithm SomeAlgo {
  variables
    output = ..., ...
  {
    \* Preconditions
    assert Q;

    (* Implementation in PlusCal *)
    Imp.

    \* Postcondition 1
    assert R1;
    \* Postcondition 2
    assert R2;
  }
}
```

Contracts

assess: (1) given valid input, your Q must pass. (2) given invalid input, your Q must fail.

(pre-state)

(post-state) may involve both pre- and post-state values.

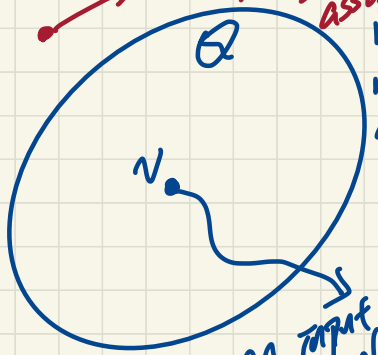
Q complete?

– a predicate which states the assumption on input value(s) made by alg.

to be passed to the alg

an input value sat. Q is not assumed

Q must be mapped, by alg., to the correct output val.



an input value satisfying Q

Correct Algorithm and Complete Postcondition (1.2)

```
------- MODULE ExampleModule -------
EXTENDS Integers, Sequences, TLC
CONSTANT inputSeq, inputVal
-- algorithm BinarySearch {
  variables
    output = FALSE, ...
  {
    \* Preconditions
    assert Q;

    (* Implementation in PlusCal *)
    Imp.

    \* Postcondition 1
    assert R1;
    \* Postcondition 2
    assert R2;
  }
}
```

Q complete?

Extend?

inputSeq is sorted

(1) Passing <1, 2, 3, 4>

should not trigger any precond errors.

(2) Passing <1, 3, 2, -2>

should trigger some precond. error.

Correct Algorithm and Complete Postcondition (2.1)

```
------- MODULE ExampleModule -------
EXTENDS Integers, Sequences, TLC
CONSTANT input
-- algorithm SomeAlgo {
  variables
    output = ..., ...
  {
    \* Preconditions
    assert Q;

    (* Implementation in PlusCal *)
    Imp.

    \* Postcondition 1
    assert R1;
    \* Postcondition 2
    assert R2;
  }
}
```

2 General

Specify one or more predicate assertions relating the input and output variables.

→ Postconditions do not need to adapt for different input values

Imp. correct?

Given that input satisfies precond Q,
does the alg. terminate in a state where the output has the expected val.

Two ways for asserting the output:

1. Specific: alg(input1) = expO1 (actual output)
   alg(input2) = expO2

drawback: assert expO1 has to be changed according to the specific input.

Correct Algorithm and Complete Postcondition (2.2)

```
------- MODULE ExampleModule -------
EXTENDS Integers, Sequences, TLC
CONSTANT inputSeq, inputVal
(* algorithm BinarySearch {
  variables
    output = FALSE, ...
  {
    \* Preconditions
    assert /* inputSeq is sorted */;

    (* Implementation in PlusCal *)
    Imp.

    \* Postcondition 1
    assert /* inputSeq unchanged */;
    \* Postcondition 2
    assert /* output computed correctly */;
  }
}
```

output = inputVal ∈ inputSeq.

Imp. correct?

[1] Specific

sorting

set input = <2,4,1,3>

assert output = <1,2,3,4>

search

inputSeq = <1,2,3,4>

inputVal = 5

assert output = FALSE

[2] General

↓

specify the "complete" set of postconditions to be checked.

Correct Algorithm and Complete Postcondition (3.1)

```
------- MODULE ExampleModule -------
EXTENDS Integers, Sequences, TLC
CONSTANT input
(* algorithm SomeAlgo {
  variables
    output = ..., ...
  {
    \* Preconditions
    assert Q;

    (* Implementation in PlusCal *)
    Imp.

    \* Postcondition 1
    assert R1;
    \* Postcondition 2
    assert R2;
  }
}
```

say you're required to specify each one of them.

R1 and R2 complete?

(1) if the kinds of imp. errors are captured by either one of them

(2) if they account for all possible ways that the imp. can go wrong.

Correct Algorithm and Complete Postcondition (3.2)

```
------- MODULE ExampleModule -------
EXTENDS Integers, Sequences, TLC
CONSTANT inputSeq, inputVal
(* algorithm BinarySearch {
  variables
    output = FALSE, ...
  {
    \* Preconditions
    assert /* inputSeq is sorted */;

    (* Implementation in PlusCal *)
    Imp.

    \* Postcondition 1
    assert R1;
    \* Postcondition 2
    assert R2;
  }
}
```

unchanged input

correct output

answers submitted by you.

R1 and R2 complete?

(1) Assess quality of your R1?

Replace by a faulty imp. that always removes the 1st item in input.

(2) Assess quality of your R2?

Replace by a faulty imp. which (a) does not modify input (b) always returns false

Predicate Logic: Exercise 1

Consider the following predicate:

$$\forall\, x, y \bullet x \in \mathbb{N} \land y \in \mathbb{N} \Rightarrow x * y > 0$$

→ not a theorem.

Choose **all** statements that are **correct**.

1. It is a theorem, provable by (5, 4). ✗
2. It is a theorem, provable by (2, 3). ✗
3. ✓ It is not a theorem, witnessed by (5, 0).
4. It is not a theorem, witnessed by (12, -2).
5. It is not a theorem, witnessed by (12, 13).

$5 \in \mathbb{N} \land 0 \in \mathbb{N} \Rightarrow 5 * 0 > 0$

T ⇒ F