

**Assignment 3: Program with Loops and Arrays**  
**15-414/15-424 Bug Catching: Automated Program Verification**

Due: **11:59pm**, Sunday 2/23/20

Total Points: 50

1. **Unsoundness of while invariants (10 points)** Consider the following modified formulation of the while loop rule (changes highlighted in bold):

$$(R1) \quad \frac{\Gamma \vdash J, \Delta \quad J, Q \vdash [\alpha]J \quad J, \neg Q \vdash P, \mathbf{\Delta}}{\Gamma \vdash [\mathbf{while}(Q) \alpha]P, \Delta}$$

- Show that rule R1 is unsound. That is, give an instance of rule R1 with concrete formulas in which all premises are valid but the conclusion is not valid.
- Briefly explain why that happens, and why the sound rule discussed in lecture does not allow it.

## 2. Repeat until... (5 points)

Aside from while loops, many other flavors of imperative looping construct have been proposed and implemented, among them the repeat-until loop:

`repeat  $\alpha$  until  $Q$`

As the syntax suggests, unlike the while loop, repeat-until begins executing its body  $\alpha$ , and continues doing so until a condition  $Q$  is met.

- Define the semantics of this command, either by explicitly defining the set of initial and final states it realizes, or by defining the command in terms of others that we have discussed this semester.
- Explain in words how your semantics fit the intuitive description of the command's behavior.

*Hint: read the next question before completing this, as your choice of semantics may make a difference.*

### 3. Repeated soundness (15 points)

Now provide an inference rule for repeat-until, and prove its soundness.

$$(R2) \frac{\dots}{\Gamma \vdash [\text{repeat } \alpha \text{ until } Q]P, \Delta}$$

- As with the rule for while loops discussed in lecture, your rule should make use of an invariant  $J$  that is preserved by the body and used (possibly with other facts) to prove the postcondition.
- You may prove soundness either by direct semantic argument, or if possible, by formal sequent derivation.

#### 4. Reads of writes (10 points)

For each of the formulas below, determine whether the formula involving arrays is valid or not. If the formula is valid, provide a sequent proof. If not, give a counterexample and brief explanation.

To get you started, suppose that the formula in question were  $a\{i \mapsto e\}(j) = e \rightarrow i = j$ . This formula is not valid, and a counterexample would be given by the following array and variable assignments:  $a = \{0 \mapsto 0\}$  with  $i = 1, e = 0, j = 0$ . This counterexample falsifies the formula above, because  $a\{1 \mapsto 0\}(0) = 0$  but  $i = 1 \neq 0 = j$ .

(a)  $a\{i \mapsto e\}(j) = e \rightarrow a(j) = e$

(b)  $a(k) = e' \wedge j \neq k \wedge i = j \rightarrow [a(i) := e; a(j) := \tilde{e}]a(k) = e'$

5. **Wildcard revisited (10 points)** Recall from the previous homework designing an axiom for the wildcard assignment command  $x := *$ . In the event that you want to prove a diamond formula involving this command, you would need an analogous axiom to go along with it:

$$(\langle := * \rangle) \quad \langle x := * \rangle p(x) \leftrightarrow \dots$$

State an axiom  $\langle := * \rangle$ , and prove that it is sound. *Hint: your proof might be simplified by making use of relationships that must hold between boxes and diamonds.*