Assignment 3
Dynamic Duo

15-414: Bug Catching: Automated Program Verification

Due 23:59pm, Thursday, Feb 17, 2022
75 pts

This assignment is due on the above date and it must be submitted electronically on Gradescope. Please carefully read the policies on collaboration and credit on the course web pages at http://www.cs.cmu.edu/~15414/s22/assignments.html.

What To Hand In

You should hand in the following files on Gradescope:

- Submit the file `asst3.zip` to Assignment 3 (Code). You can generate this file by running `make handin`. This will include your solutions `partition.mlw` and the proof session in `partition/`.
- Submit a PDF containing your answers to the written questions to Assignment 3 (Written). You may use the file `asst3.tex` as a template and submit `asst3.pdf`.

  Make sure your session directories and your PDF solution files are up to date before you create the handin file.

Using LaTeX

We prefer the answer to your written questions to be typeset in LaTeX, but as long as you hand in a readable PDF with your solutions it is not a requirement. We package the assignment source `asst3.tex` and a solution template `asst3-sol.tex` in the handout to get you started on this.

1 Lather, Rinse, Repeat (15 pts)

In this problem we continue to study a *repeat-until* loop as an alternative to a *while* loop in our DL language. Informally, the $\mathsf{repeat}\,\alpha\,P$ loop executes α and then tests P. If P is true it exits the loop, and if P is false it repeats it.

Task 1 (5 pts). The most straightforward (but relatively difficult to use) axiom for while loops in dynamic logic is $[\mathsf{while}\,P\,\alpha]Q \leftrightarrow (P \to [\alpha][\mathsf{while}\,P\,\alpha]Q) \wedge (\neg P \to Q)$. Give a corresponding axiom for the repeat loop.

Task 2 (5 pts). Express the repeat-until loop using the constructs of nondeterministic dynamic logic where the conditional and while loop have been replaced by nondeterministic choice and repetition.

Task 3 (5 pts). Provide an axiom for reasoning with loop invariants for repeat-until loops

$$[\mathsf{repeat}\,\alpha\,P]Q \leftarrow \ldots$$

where the right-hand side only refers to $[\alpha]\_$, $\Box\_$, P, Q, and a loop invariant J.

2 Looking into the Past (25 pts)

In ordinary modal logic there is a $\blacksquare P$ modality that expresses "*P has always been true*". We can extend dynamic logic with a corresponding operator $(\!|\alpha|\!)P$ read as "*before α P*". Its semantics is defined by

$$\omega \models (\!|\alpha|\!)P \quad \text{iff for all } \mu \text{ such that } \mu[\![\alpha]\!]\omega \text{ we have } \mu \models P$$

For each of the following parts, develop axioms for nondeterministic dynamic logic that allow you to break down proving $(\!|\alpha|\!)P$ into properties of smaller programs or eliminate them altogether. You only need to prove one direction of one of these properties (see Task 8) but it may be helpful to convince yourself your answers are correct.

Task 4 (5 pts). $(\!|\alpha\,;\,\beta|\!)P$

Task 5 (5 pts). $(\!|\alpha \cup \beta|\!)P$

Task 6 (5 pts). $(\!|?Q|\!)P$

Task 7 (5 pts). $(\!|\alpha^*|\!)P$. In this task, both sides can refer to α^*.

Task 8 (5 pts). Prove one direction of one of the axioms from Tasks 4–7. For this purpose assume $\omega \models \mathsf{ONESIDE}$ and prove that $\omega \models \mathsf{OTHERSIDE}$ for an arbitrary ω. Since ω is arbitrary this means that the implication is valid. The proof regarding sequential composition in Lecture 6, Section 5 provides a good model for the format and level of detail we expect.

3 Partition Party (15 pts)

This problem exercises the often tricky aspects of modifying a data structure in place—in this case a simple array of integers.

Write and verify a function `partition (a : array int) : int` that permutes the elements of the array `a` in place so that all negative numbers precede all nonnegative numbers. The value returned is the index of the first nonnegative number in the resulting array, or `a.length` if the numbers are all negative.

You can find a solution template in file `partition.mlw`.

Hint: the standard libraries `array.ArrayPermut` and `array.ArraySwap` may be helpful.

4 The Day of Judgment (20 pts)

Task 9 (12 pts). For each of the following judgments in dynamic logic, find a program that, when substituted for α, makes the judgment hold. Throughout these judgments, ω is an arbitrary state.

1. $\omega[x \mapsto 0, y \mapsto 42] \models [(?(x \neq y); x \leftarrow x + 1; \alpha)^*; (?(x = y))]\bot$
2. $\omega[x \mapsto 0, y \mapsto 0] \models ([\alpha](x = 42)) \wedge ([\alpha; \alpha](x = 42)) \wedge ([\alpha; \alpha; \alpha](x \neq 42))$
3. $\omega[x \mapsto 0, y \mapsto 0] \models \neg[\alpha](x \neq y \vee \langle\alpha\rangle(x = y))$
4. $\omega[x \mapsto 0, y \mapsto 0] \models (\langle\alpha\rangle(x = 42)) \wedge (\neg[\alpha](x = 42))$

Task 10 (8 pts). For each of the following judgments in dynamic logic, find a state that, when substituted for ω, makes the judgment hold. Here, $\mathsf{skip} \triangleq ?\mathsf{true}$.

1. $\omega \models [(?(x \neq y); x \leftarrow x + 1; y \leftarrow y - 1)^*; (?(x = y))](x \neq y)$
2. $\omega \models \neg[(x \leftarrow x + 1; y \leftarrow 2x)^*](x = y)$
3. $\omega \models [\mathsf{if}\ (x = 0)\ (y \leftarrow y * 0)\ (\mathsf{skip})](x = 0 \rightarrow y = 42)$