MP 3 – Hoare Logic Proofs in Isabelle/HOL

CS 477 – Spring 2018
Revision 1.0

Assigned March 15, 2018
Due March 28, 2018, 9:00 PM
Extension extend48 hours (penalty 20% of total points possible)

1 Change Log

1.0 Initial Release.

2 Objectives and Background

The purpose of this MP is to test the student's ability to

- to write proofs using a "lifted" version of propositional logic, prediciate logic, and basic arithmetic in Isabelle;
- write simple proofs in Hoare Logic using its encoding in Isabelle,
- see the effects of using verification condition generation and weakest precondition calculation to simplify Hoare Logic proofs..

Another purpose of MPs in general is to provide a framework to study for the exam. Several of the questions on the exam will appear similar to the MP problems.

3 Turn-In Procedure

A skeleton version of the file `mp3.thy` for this assignment should be found in the `assignments/mp3/` subdirectory of your svn directory for this course. You should put code answering each of the problems below in the file `mp3.thy`. Your completed `mp3.thy` file should be put in the `assignments/mp3/` subdirectory of your svn directory (where it was originally found) and committed as follows:

```
svn commit -m "Turning in mp3"
```

Please read the *Instructions for Submitting Assignments* in

`http://courses.engr.illinois.edu/cs477/mps/index.html`

You may find it helpful to refer to Chapters 2, 4 and 12 of Concrete Symantics, which you can find in you svn repository at `resources/concrete_semantics.pdf`.

4 Problems

The problems below are designed to step you through some of the pieces of reasoning about Hoare Logic proofs in Isabelle. The first set give you practice using a combination of "lifted" rules and facts from propositional logic, and the definitions of "lifted" operators to reduce the problem to tractable problems in basic HOL. The second set extend this exercise to reasoning about "lifted" arithmetic facts, and the third set ask you to prove certain Hoare triples.

4.1 Lifted Propositional Logic

In the first three problems below you will prove the "lifted" versions of three problems from MP1. You are free to use any and all theorem proving methods in Isabelle to prove them. You may wish to refer to the definitions and theorems in `lifted_basic` and `lifted_predicate_logic`. For an example, here is the "lifted" version of the first problem from MP1:

lemma *Mp1-problem1*: $|\models((A\ [\wedge]\ B)\ [\longrightarrow]\ (B\ [\wedge]\ A))$
apply (*rule bvalid-imp-bI*)
by (*simp add*: *and-b-def*)

Remove the `oops` from each problem and put in your own proof.

1. (5 pts)

lemma *problem1*: $|\models((A\ [\wedge]\ B)\ [\longrightarrow]\ (([\neg]B)\ [\longrightarrow]\ ([\neg]A)))$
oops

2. (5 pts)

lemma *problem2*: $|\models((A\ [\longrightarrow]\ B)\ [\longrightarrow]\ (([\neg]\ B)\ [\longrightarrow]\ ([\neg]\ A)))$
oops

3. (5 pts)

lemma *problem3*: $|\models(([\neg]A\ [\vee]\ [\neg]B)\ [\longrightarrow]\ ([\neg](A\ [\wedge]\ B)))$
oops

4.2 Lifted Arithmetic Facts

For the next three problems, you may additionally wish to use definitions and theorems in `lifted_int_data` as well as those in `lifted_predicate_logic`. In addition to `simp`, `clarsimp`, and `auto`, you may find `arith` sometimes useful, once you have reduced the problem to "ordinary" arithmetic. It solves problems in linear arithmetic (no multiplying variables together). Alternately, if you have reduced your problem to "ordinary" arithmetic, you will probably find that Sledgehammer (the third tab in the list at the bottom) generally can find a proof for you. If you get stuck, don't waste time thrashing; ask for help on piazza or in my office.

Depending on how you attack these problems, you may be faced with showing two functions taking states as arguments are equal. You can turn this into a "ground" equality between to values with

```
apply (rule ext)
```

The following is an example that was worked in class as a part of a larger Hoare Logic Problem:

lemma *arith-example*:
$|\models$ (($\$\ ''p''\ [=]\ \$\ ''i''\ [\times]\ \$\ ''y''\ [\wedge]\ \$\ ''i''\ [\leq]\ (\$\ ''x'')\ [\wedge]$
$\$\ ''i''\ [<]\ \$\ ''x'')\ [\longrightarrow]$
$(\$\ ''p''\ [+]\ \$\ ''y''\ [=]\ (\$\ ''i''\ [+]\ k\ 1)\ [\times]\ \$\ ''y''\ [\wedge]$
$\$\ ''i''\ [+]\ k\ 1\ [\leq]\ \$\ ''x'')$)
apply (*simp add*: *eq-b-def imp-b-def plus-e-def and-b-def less-b-def*
less-eq-b-def times-e-def k-def rev-app-def bvalid-def)
by (*simp add*: *distrib-right*)

4. (7 pts)

lemma *problem4*:
$|\models(\$''x''\ [\times]\ (\$''y''\ [+]\ \$''z'')\ [=]\ ((\$''x''\ [\times]\ \$''y'')\ [+]\ (\$''x''\ [\times]\$''z'')))$
oops

5. (6 pts)

lemma *problem5*:
$|\models$ ((($\$\ ''y''\ [+]\ \$\ ''x''\ [=]\ k\ (a + b)$) $[\wedge]$
$(\$\ ''x''\ [\geq]\ k\ 0)\ [\wedge]\ ([\neg](\$\ ''x''\ [>]\ k\ 0)))\ [\longrightarrow]$
$(\$\ ''y''\ [=]\ k\ (a + b)))$

`oops`

6. (8 pts)

lemma *problem6*:
$|\models$ ((($\$\ ''y''\ [=]\ \$\ ''a'')\ [\wedge]\ ([\neg](\$\ ''y''\ [mod]\ k\ 2\ [=]\ k\ 0)))\ [\longrightarrow]$
$((\$\ ''y''\ [+]\ k\ 1\ [\geq]\ \$\ ''a'')\ [\wedge]\ (\$\ ''y''\ [+]\ k\ 1\ [\leq]\ \$\ ''a''\ [+]\ k\ 1)\ [\wedge]$
$((\$\ ''y''\ [+]\ k\ 1)\ [mod]\ k\ 2\ [=]\ k\ 0)))$
oops

4.3 Hoare Logic Proofs

In the next set of problems, you will want to rely upon the rules in `Hoare_SIMP` that define the relation `hprovable` describing which Hoare triples are provable. If you want Isabelle to figure out some of the missing pieces for you, you may wish to alter the order in which you solve the subgoals. You may use `prefer` n to select the n^{th} subgoal as the next one you will work on.

7. (15 pts)

lemma *problem7*:
$\{\{\$''y''\ [=]\ \$''a''\ \}\}$
IF $\$''y''\ [mod]\ (k\ 2)\ [=]\ (k\ 0)$ *THEN* $(''y''\ ::=\ \$''y'')$ *ELSE* $(''y''\ ::=\ \$''y''\ [+]\ (k\ 1))$ *FI*
$\{\{(\$''y''\ [\geq]\ \$''a'')\ [\wedge]$
$(\$''y''\ [\leq]\ (\$''a''\ [+]\ (k\ 1)))\ [\wedge]$
$(\$''y''\ [mod]\ (k\ 2)\ [=]\ k\ 0)\}\}$
oops

8. (21 pts)

lemma *problem8*:
{{$ ″y″ [=] k a [∧] $ ″x″ [=] k b [∧] k b [>] k 0}}
WHILE $ ″x″ [>] k 0 *DO*
(″y″ ::= $ ″y″ [+] k 1;;
 ″x″ ::= $ ″x″ [−] k 1)
OD
{{$ ″y″ [=] k (a + b)}}
oops

4.4 Verification Conditions

Lastly, I ask you to redo `problem8`, but this time where you use none of the rules defining `hprovable`, or the two derived rules of consequence, but instead make use of the rules `alt_ann_provable_provable`, `vcg_and_P` found in `Hoare_ann_SIMP`. After that, `simp` will remove the instances of `vcg` and `wp`, and you are left with a problem like the ones in Subsection 4.2.

9. (10 pts)

lemma *problem9*:
{{$ ″y″ [=] k a [∧] $ ″x″ [=] k b [∧] k b [>] k 0}}
WHILE $ ″x″ [>] k 0 *DO*
(″y″ ::= $ ″y″ [+] k 1;;
 ″x″ ::= $ ″x″ [−] k 1)
OD
{{$ ″y″ [=] k (a + b)}}
oops

4.5 Extra Credit

10. (10 pts)

lemma *problem10*:
{{$ ″n″ [=] k a [∧] k a [>] k 0}}
″r″ ::= k 0;;
WHILE ($ ″n″ [>] k 0) *DO*
″r″ ::= $ ″r″ [+] $ ″n″;;
″n″ ::= $ ″n″ [−] k 1
OD
{{k 2 [×] $ ″y″ [=] k (a ∗ (a + 1))}}
oops