Quiz

Synthesis, Analysis, and Verification 2012

Tuesday, April 17th, 2012

Last Name : _____

First Name : _____

Exercise	Points	Achieved Points
1	10	
2	10	
3	20	
4	30	
5	15	
6	15	
Total	100	

Problem 1: Relations (10 points)

In this quiz we use the following notation:

- S is a set of all states
- $\mathsf{sp} : 2^S \times 2^{S \times S} \to 2^S$ is defined by

$$\mathsf{sp}(P, r) = \{s' \mid \exists s.s \in P \wedge (s, s') \in r\}$$

- $\mathsf{wp} : 2^{S \times S} \times 2^S \to 2^S$ is defined by

$$\mathsf{wp}(r, Q) = \{s \mid \forall s'.(s, s') \in r \to s' \in Q\}$$

- Diagonal relation Δ_A for $A \subseteq S$ and $r \subseteq S \times S$ is defined by

$$\Delta_A = \{(s, s) \mid s \in A\}$$

Task a) Prove that for every $P \subseteq S$,

$$sp(P, r) = sp(S, \Delta_P \circ r)$$

Task b) Prove that for every $P \subseteq S$,

$$wp(r, P) = \{x | sp(\{x\}, r) \subseteq P\}$$

Problem 2: Fixpoints (10 points)

Let $r \subseteq A \times A$.

Task a) Let $D = \{r \mid r \subseteq A \times A\}$ and let $F_r : D \to D$ be defined as

$$F_r(s) = \Delta_A \cup s \circ r \circ s$$

Compute the least fixpoint of F according to the subset relation $\subseteq$ and give an alternative short description of it.

Task b) Now let

$$F_r(s) = s \circ r \circ s$$

What is the least fixpoint now?

Problem 3: Nice Invariants (20 points)

Suppose that $R(v, v')$ is a formula describing a piece of code and let $r = \{(v, v') \mid R(v, v')\}$ be the relation corresponding to R. Let $T(v, v')$ be a formula such that, for $t = \{(v, v') \mid T(v, v')\}$, the following holds:

$$\Delta \subseteq t$$
$$r \subseteq t$$
$$t \circ t \subseteq t$$

Let $P(v)$ and $Q(v)$ be formulas and assume that the following formula is valid:

$$P(v) \wedge T(v, v') \rightarrow Q(v')$$

We will say that $I(v)$ is a *nice invariant* if all of the following formulas are valid:

$$P(v) \rightarrow I(v)$$
$$I(v) \rightarrow Q(v)$$
$$I(v) \wedge R(v, v') \rightarrow I(v')$$

Task a) Prove or disprove that $sp(P, t)$ is a nice invariant.

Task b) Prove or disprove that $wp(t, Q)$ is a nice invariant.

Task c) How do the answers in a) and b) change if we replace the condition $t \circ t \subseteq t$ with the condition $r \circ t \subseteq t$? Recall that we define relation composition $\circ$ such that $r \circ t = \{(v, v') \mid \exists v''.(v, v'') \in r \wedge (v'', v') \in t\}$

Problem 4: Hoare Triples and Loop Invariants (30 points)

Consider a programming language that supports integer variables, as well as variables that denote sets of integers and binary relations on integers (all integers are unbounded).
The command $\mathsf{lookup}(k, r)$ looks up a value v such that $(k, v) \in r$. If such value exists, it returns one such value as a singleton set $\{v\}$. If no such value exists, it returns the emptyset $\{\}$. (Note that, for each k, there can in general be zero, one, or more values v such that $(k, v) \in r$.)
Task a) Write a Hoare triple describing $\mathsf{lookup}(k, r1)$ in the form

$$\{\mathit{precondition}\} \;\; v1 = \mathsf{lookup}(k, r1) \;\; \{\mathit{postcondition}\}$$

where the precondition is as permissive (weak) as possible (so that it does not restrict the application of the lookup operation unnecessarily). Given as week precondition as you can find, specify the most precise postcondition that follows from the above description of how lookup should work.

Task b) Consider the following program, where the variables r1,r are relations, v1,W are sets of integers, and k is an integer.

```
// Precondition: ∀i.∀v.(i,v) ∈ r → 0 ≤ i
r1 = r;
k = 0;
W = {};
while // invariant Inv
        (r1 != {})
{
  v1 = lookup(k,r1);
  if (v1 = {}) {
    k = k + 1
  } else {
    W = W ∪ v1;
    r1 = r1 \ ({k} × v1)
  }
}
// Postcondition: W = range(r)
```

We use the notation

$$\mathsf{range}(r) = \{v \mid \exists i.(i, v) \in r\}$$

Find an appropriate loop invariant, Inv, and use it to prove that, whenever we run the above program in a state that satisfies the Precondition, its final state satisfies the Postcondition. You need to explain why (1) the invariant holds initially in *all* states that satisfy the precondition, why (2) it is inductive (preserved on each execution of the loop body starting from *any* state satisfying only the invariant), and why (3) it can be used to prove the Postcondition. State each of these conditions as a Hoare triple, and prove it. Your proof of individual Hoare triples need not be very detailed.
Feel free to use any notation of sets, relations, and quantifiers in your invariants and Hoare triples. It is crucial that your invariant is correct (conditions (1),(2),(3) hold). Hint: using $r \setminus r_1$ as part of your loop invariant may be helpful.
Bonus Task c) Suppose that we modify the code above by inserting the assignment command 'k = k + 1' also in the second branch of 'if'. Does your original invariant still apply to the modified program?

Problem 5: Interval Analysis (15 points)

Consider interval analysis of a program with two integer variables x and y.
The state of the program is a map of the form $\{x \mapsto i, y \mapsto j\}$ where $i, j \in \mathbb{Z}$.
The abstract domain A associates an interval to each variable: it is the set of maps of the form

$$\{x \mapsto [l_x, u_x], y \mapsto [l_y, u_y]\}$$

with $l_x, l_y \in \mathbb{Z} \cup \{-\infty\}$ and $u_x, u_y \in \mathbb{Z} \cup \{\infty\}$.
The abstraction function α is such that given a set of concrete states S, $\alpha(S)(x)$ is the most precise interval containing all values of x found in S and $\alpha(S)(y)$ is the most precise interval containing all values of y found in S.
The concretization function γ is such that

$$\gamma(\{x \mapsto [l_x, u_x], y \mapsto [l_y, u_y]\}) = \{s.s(x) \in [l_x, u_x] \wedge s(y) \in [l_y, u_y]\}$$

Finally we define

$$sp^\sharp(a, c) = \alpha(sp(\gamma(a), c))$$

In the following questions the abstract postconditions need to be computed with respect to an arbitrary abstract precondition represented by $\{x \mapsto [l_x, u_x], y \mapsto [l_y, u_y]\}$. Please make sure that the abstract postconditions you give are as precise as possible.

Task a) Give the abstract strongest postcondition for each of the following statements

i) `y = 5*x^2 - 26*x +5`

ii) `x = x*y`

iii) `x = a*x + b*y`

You don't need to prove that it's the strongest.
Task b) Use the rules you determined above to compute the abstract postcondition for the following program:

```
y = 5*x - 1;
x = x - 5;
y = y*x;
```

You don't need to prove that your abstract postcondition is correct.

Problem 6: Predicate Abstraction (15 points)

Consider the set of predicates

$$\mathcal{P} = \{\text{false}, 0 \leq x, 0 \leq y, x \leq y\}$$

Let $A = 2^{\mathcal{P}}$. The meaning of a set of predicates $a \in A$, denoted $\gamma(a)$ is, as usual, the set of states that satisfies the conjunction of all predicates in a.
The precise semantics of a command `cmd` is the relation associated with `cmd`. For example, the precise semantics of the command

```
x = y; y = x + 1
```

in a program with two variables is the relation

$$\{((x, y), (x', y')) \mid x' = y \wedge y' = y + 1\}$$

For a given command `cmd` whose precise semantics is given by a relation r, let $sp^{\#}(a, \texttt{cmd})$ denote the least element $a' \in A$ such that $sp(\gamma(a), r) \subseteq \gamma(a')$.
As usual in programming languages, let `x++` denote a command that increments an integer variable `x` by 1 (assume that integer variables are unbounded).
Let $a_0 = \{0 \leq x, 0 \leq y, x \leq y\}$.
Compute the following values:

a) $\mathsf{sp}^{\#}(a_0, \texttt{x++})$

b) $\mathsf{sp}^{\#}(\mathsf{sp}^{\#}(a_0, \texttt{x++}), \texttt{y++})$

c) $\mathsf{sp}^{\#}(a_0, (\texttt{x++}; \texttt{y++}))$