Programming Languages Comprehensive Exam

August, 1998

Instructions

Answer all questions; you have four hours.

To maximize partial credit for an incomplete proof, you should:

- provide as much of the proof as you can;
- describe what remains to be proven and how that (when combined with what you have proven) would complete the proof;
- indicate (where possible) how you suspect that the missing piece *could* be proven.

The last two pages of this exam include a summary of relevant terminology and partial-correctness inference rules.

Questions

1. Prove the following law of program equivalence by using the valuation for local-definition blocks:

$$\textbf{let } \iota \textbf{ be } P \textbf{ in } (\textbf{let } \iota \textbf{ be } Q \textbf{ in } R) \;\equiv\; \textbf{let } \iota \textbf{ be } (\textbf{let } \iota \textbf{ be } P \textbf{ in } Q) \textbf{ in } R$$

That is, show that

$$[\![\textbf{let } \iota \textbf{ be } P \textbf{ in } \; (\textbf{let } \iota \textbf{ be } Q \textbf{ in } R)]\!]u = [\![\textbf{let } \iota \textbf{ be } (\textbf{let } \iota \textbf{ be } P \textbf{ in } Q) \textbf{ in } R]\!]u$$

for all u. Be sure to show all steps of your computation.

2. Prove (using the valuation for local-definition blocks) or disprove (by providing *and explaining* a counterexample) the following potential law of program equivalence:

$$\textbf{let } \iota \textbf{ be } P \textbf{ in } (\textbf{let } \iota' \textbf{ be } Q \textbf{ in } R) \;\equiv\; \textbf{let } \iota' \textbf{ be } (\textbf{let } \iota \textbf{ be } P \textbf{ in } Q) \textbf{ in } (\textbf{let } \iota \textbf{ be } P \textbf{ in } R)$$

3. Let $f, g : D \to D$ be continuous functions over a domain $(D, \sqsubseteq)$.

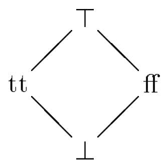
   (a) Prove that $f(\mathit{fix}(g \circ f)) = \mathit{fix}(f \circ g)$.

   (b) An element $x \in D$ is a *least common fixed point* of f and g if it is a fixed point of both f and g and it is least among all such points. That is, the *least common fixed point* of f and g (when it exists) is an element $x \in D$ such that:

   - $fx = x$ and $gx = x$;
   - for all y such that $fy = y$ and $gy = y$, $x \sqsubseteq y$.

   Prove that, if $f \circ g = g \circ f$, then f and g have a least common fixed point.

   (c) Let $(D, \sqsubseteq)$ be the domain given by $D = \{\bot, \mathrm{tt}, \mathrm{ff}, \top\}$ ordered as follows:



   Give a pair of continuous functions $f, g : D \to D$ such that $f \circ g = g \circ f$ and the least common fixed point of f and g is different from the least fixed point of f and the least fixed point of g.

   (d) Using $(D, \sqsubseteq)$ from part (c), show that the converse to part (b) is not true. That is, give a pair of continuous functions $f, g : D \to D$ such that f and g have a least common fixed point but $f \circ g \neq g \circ f$.

4. The construct *repeat C until B* provides a looping mechanism in which the loop's body is always executed at least once. In particular, the construct satisfies the following program equivalence:

$$\textit{repeat } C \textit{ until } B \;\equiv\; C;\ \textit{if } B \textit{ then skip else } (\textit{repeat } C \textit{ until } B)$$

(a) Explain how you would define the semantic valuation $[\![\textit{repeat } C \textit{ until } B]\!]$ *compositionally.* You can state your semantics in English rather than using mathematical notation if you prefer, but your answer should provide enough detail to make explicit your intended semantics and its compositional nature.

(b) Give a valid axiom (in the style of Tennent) suitable for reasoning about the partial correctness of programs including this construct.

(c) Use your axiom (along with the axioms on the attached sheet) to provide a rigorous proof of the validity of the following partial-correctness Hoare formula:

$\{y \geq 0\}$
z := 0; w := x; k := y;
repeat
    if even(k)
      then (k := k div 2; w := w + w)
      else (k := k - 1; z := z + w)
until $k = 0$
$\{z = x * y\}$

You may use without proof any facts about the integers, as well as the following derived law:

$$\{[\cdots[[P](\iota_n \mapsto E_n)]\cdots](\iota_1 \mapsto E_1)\}\,\iota_1 := E_1;\cdots;\iota_n := E_n)\{P\}$$

Terminology

A *domain* (technically, an ω-complete partial order) is a pair $(D, \sqsubseteq)$ such that:

- $\sqsubseteq$ is a partial order
- D has a least element $\perp_D$
- Every chain $x_0 \sqsubseteq x_1 \sqsubseteq \cdots \sqsubseteq x_n \sqsubseteq \cdots$ in D has a supremum (least upper bound).

A function $f : D \to E$ between domains $(D, \sqsubseteq_D)$ and $(E, \sqsubseteq_E)$ is *continuous* if, for all chains in D,

$$f(\bigsqcup_{n=0}^{\infty} x_n) = \bigsqcup_{n=0}^{\infty} f(x_n).$$

A *fixed point* of a function $f : D \to D$ is an element $x \in D$ such that $f(x) = x$. A *least fixed point* of f is a fixed point x such that, for all other fixed points y, $x \sqsubseteq y$.

The least fixed point of a continuous function f, denoted $\mathit{fix}(f)$, can be calculated by

$$\mathit{fix}(f) = \bigsqcup_{n=0}^{\infty} f^n(\perp).$$

Axioms for Floyd-Hoare Logic

These axioms are taken directly from Tennent's *Semantics of Programming Languages.*

Command-specific axioms

$\{P\} \mathit{skip} \{P\}$

$\{\mathsf{true}\} \mathit{diverge} \{\mathsf{false}\}$

$\{[P](\iota \mapsto E)\} \iota := E \{P\}$

$\{P\} C \{Q\} \Longrightarrow \{P\}(C)\{Q\}$

$\{P\} C_0 \{Q\}$ & $\{Q\} C_1 \{R\} \Longrightarrow \{P\} C_0; C_1 \{R\}$

$\{P \text{ and } B\} C_0 \{Q\}$ & $\{P \text{ and not } B\} C_1 \{Q\} \Longrightarrow \{P\} \mathit{if}\ B\ \mathit{then}\ C_0\ \mathit{else} C_1 \{Q\}$

$\{P \text{ and } B\} C \{P\} \Longrightarrow \{P\} \mathit{while}\ B\ \mathit{do}\ C \{P \text{ and not } B\}$

Other axioms

Pre-condition Strengthening:

$P_0 \supset P_1$ & $\{P_1\} C \{Q\} \Longrightarrow \{P_0\} C \{Q\}$

Post-condition Weakening:

$Q_0 \supset Q_1$ & $\{P\} C \{Q_0\} \Longrightarrow \{P\} C \{Q_1\}$

Pre-condition Disjunction:

$\{P_1\} C \{Q\}$ & $\cdots$ & $\{P_n\} C \{Q\} \Longrightarrow \{P_1 \text{ or } \cdots \text{or } P_n\} C \{Q\}$

Post-condition Conjunction:

$\{P\} C \{Q_1\}$ & $\cdots$ & $\{P\} C \{Q_n\} \Longrightarrow \{P\} C \{Q_1 \text{ and } \cdots \text{and } Q_n\}$