Semantics of Programming Languages Comprehensive Exam

January, 1999

Instructions

Answer all parts of 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

1a) Is the following equivalence valid?

$$(\textbf{for}\ \mathrm{N}\ \textbf{do}\ \mathrm{N} := \mathrm{N}{+}1) \equiv \textbf{diverge}$$

If the equivalence is valid, prove it. If it is invalid, give a couner-example and explain why your counter-example shows the equivalence is invalid.

1b) Is the following equivalence valid?

$$(\textbf{if}\ \mathrm{B}\ \textbf{then}\ \mathrm{C}_1\ \textbf{else}\ \mathrm{C}_2);\ \mathrm{C}_3 \equiv \textbf{if}\ \mathrm{B}\ \textbf{then}\ (\mathrm{C}_1;\mathrm{C}_3)\ \textbf{else}\ (\mathrm{C}_2;\mathrm{C}_3)$$

If the equivalence is valid, prove it. If it is invalid, give a couner-example and explain why your counter-example shows the equivalence is invalid.

2) Let D be a domain with partial order relation $\sqsubseteq$.

Definition: Let $f : D \longrightarrow D$ and let $d \in D$. f is *increasing at* d if $d \sqsubseteq f(d)$. An element e of D is the *least fixed point* of f *above* d if

- $d \sqsubseteq e$,
- $f(e) = e$,
- for all e' such that $d \sqsubseteq e'$: $f(e') = e'$ implies $e \sqsubseteq e'$.

Denote the least fixed point of f above d, if it exists, by $\mathit{fix}_d(f)$.

Proposition: If $f : D \longrightarrow D$ is continuous, and increasing at d, then $\mathit{fix}_d(f)$ exists and

$$\mathit{fix}_d(f) = \bigsqcup_{i=0}^{\infty} f^i(d)\ .$$

Part a) Prove the proposition.

Part b) Let $f, g : D \longrightarrow D$ be continuous functions on D that are both increasing at d. Prove that all of the fixed points in the following equation exist, and that the equation holds:

$$g(\mathit{fix}_d(f \circ g)) = \mathit{fix}_d(g \circ f)\,.$$

3) Give a formal proof in Floyd-Hoare logic (see attachment) that the following command is partially correct with respect to the indicated pre- and post-conditions.

$\{A \geq 0\}$
X := 0;
Y := A;
while X < Y **do**
  (Y := Y - 1; X := X + 1);
$\{Y = A \text{ div } 2\}$

4) Let $D_\perp = \{a, b, \perp\}$ be a domain with bottom element $\perp$. Thus, $\perp \sqsubseteq a$ and $\perp \sqsubseteq b$. Suppose that a and b are incomparable in the partial order $\sqsubseteq$. Let $S = \{x, y\}$. S is just a set without any ordering.

Part a) List the elements in the set of functions $S \longrightarrow D_\perp$ and describe the ordering among them in the "pointwise" ordering that is determined by $\sqsubseteq$. Note that each element that you must list is a function, and that a function can be described by listing the ordered pairs in its graph.

Part b) Now Let D and E be domains with bottom elements. A function $f : D \longrightarrow E$ is *strict* if $f(\perp_D) = \perp_E$. Suppose D has exactly 1000 elements and E has exactly 50 elements. How many functions in the set of functions $D \longrightarrow E$ (the set of all functions from D to E) are strict?

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 and Rules of Inference for Floyd-Hoare Logic

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

Command-specific axioms and rules of inference

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

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

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

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

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

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

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

Other axioms and rules of inference

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:

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

Post-condition Conjunction:

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

Semantics of some commands

$[\![(X)]\!] = [\![X]\!]$.

$[\![\,\textbf{if}\ B\ \textbf{then}\ X_0\ \textbf{else}\ X_1\,]\!]s = \begin{cases} [\![X_0]\!]s & \text{if } [\![B]\!] = \textit{true} \\ [\![X_1]\!]s & \text{if } [\![B]\!] = \textit{false} \end{cases}$

$[\![\textbf{true}]\!]s = \textit{true}$

$[\![\textbf{not}\ B]\!]s = \begin{cases} \textit{false} & \text{if } [\![B]\!]s = \textit{true} \\ \textit{true} & \text{if } [\![B]\!]s = \textit{false} \end{cases}$

$[\![B_0\ \textbf{and}\ B_1]\!]s = ([\![B_0]\!]s \text{ and } [\![B_1]\!]s)$

$[\![N_0 < N_1]\!]s = ([\![N_0]\!]s < [\![N_1]\!]s)$

$[\![E_0 = E_1]\!]s = ([\![E_0]\!]s < [\![E_1]\!]s)$

$[\![0]\!]s = 0$

$[\![\textbf{succ}\ N]\!]s = ([\![N]\!]s) + 1$

$[\![\textbf{skip}]\!]s = s$

$[\![C_0; C_1]\!]s = [\![C_1]\!]([\![C_0]\!]s)$

$[\![\,\textbf{for}\ N\ \textbf{do}\ C\,]\!]s = [\![C]\!]^{[\![N]\!]s}(s)$

$[\![\textbf{diverse}]\!]s$ is undefined, for all states s.