UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2001

BEng Honours Degree in Computing Part II
MEng Honours Degrees in Computing Part II
for Internal Students of the Imperial College of Science, Technology and Medicine

*This paper is also taken for the relevant examinations for the Associateship of the City and Guilds of London Institute*

# PAPER C242=MC242

# OPERATIONAL SEMANTICS

Friday 18 May 2001, 14:30
Duration: 90 minutes
(Reading time 5 minutes)

*Answer THREE questions*

Paper contains 4 questions
Calculators not required

1) The following is the abstract syntax of a Caterpillar control language:

$$p \in \mathit{Program}$$
$$p ::= \texttt{move} \mid \texttt{L} \mid \texttt{R} \mid p_1 \texttt{;} p_2$$

The Caterpillar 'moves' around on an infinitly large *board*, composed of *fields* (with coordinates), that contains a number of juicy green *leaves* (possibly zero).

- Using **move**, it can move into the next field with respect to the direction it is pointing. However, if the **move** brings the Caterpillar onto an empty field, the Caterpillar will not stop; instead, it will continue to move in the same direction until it finds a leaf.
- While it moves around, if it reaches a field with leaves in it, it will eat *one*.
- Using **L** and **R**, the direction that the Caterpillar is pointing at will change by 90°.

The state of the Caterpillar is represented by a triple: $\langle pos, dir, \texttt{F} \rangle$.

a. Specify what '*pos*', '*dir*' and '**F**' should be. It is possible to have *looping* programs? If 'yes', give an example; if 'no', explain your answer.

b. Using your answer to the previous part, give the Structural Operational Semantics of Caterpillar programs.

c. The Caterpillar Machine has configurations

$$\langle c, e, s \rangle \in \mathit{Code} \times \mathit{State},$$

where

$$c \in \mathit{Code}$$
$$i \in \mathit{Instruction}$$
$$c ::= \varepsilon \mid i : c.$$

With the intention of defining a suitable translation function to translate Caterpillar control programs into Caterpillar machine code, specify the set of instructions, and define an operational semantics for the Caterpillar Machine.

Give the intended suitable translation function to translate Caterpillar control programs into Caterpillar machine code.

d. Assuming that:

$$\text{if } \langle c_1, e_1, s \rangle \rhd^k \langle c', e', s' \rangle \text{ then } \langle c_1 : c_2, e_1 : e_2, s \rangle \rhd^k \langle c' : c_2, e' : e_2, s' \rangle,$$

show that your translation function for programs is correct with respect to the Natural Semantics. Show at least one base case and one inductive case.

The four parts carry 20%, 20%, 30%, and 30% of the marks, respectively.

2) The abstract syntax for the language Exif-Loop is given by:

$$\begin{array}{rcl} x & \in & \textit{Variables} \\ a & \in & \textit{Arithmetic Expressions} \\ b & \in & \textit{Boolean Expressions} \\ S & \in & \textit{Statements} \end{array}$$

$$\begin{array}{rcl} a & ::= & n \mid a_1 + a_2 \mid a_1 - a_2 \mid a_1 \times a_2 \\ b & ::= & \texttt{true} \mid \texttt{false} \mid (a_1 = a_2) \mid (a_1 \leq a_2) \mid (\neg b) \mid (b_1 \;\&\; b_2) \\ S & ::= & x := a \mid S_1 \texttt{;}\, S_2 \mid \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2 \mid \texttt{skip} \\ & & \mid \texttt{loop}\ S_1\ \texttt{exif}\ b\texttt{;}\, S_2\ \texttt{endl} \end{array}$$

(The idea of the **loop** − **endl** construct is that, in contrast to a **while** loop, a number of statements will be executed before testing the boolean. If the boolean tests **tt**, then execution of the loop is terminated.)

a. Assuming the existence of the functions $b \mapsto \mathcal{B}\,[\![b]\!]\,s$ and $a \mapsto \mathcal{A}\,[\![a]\!]\,s$ that define the semantics of boolean and arithmetic expressions, define the Natural Semantics for Exif-Loop.

b. Are the programs

$$\texttt{loop}\ S_1\ \texttt{exif}\ b\texttt{;}\, S_2\ \texttt{endl} \quad \text{and} \quad S_1\texttt{;}\,\texttt{loop skip exif}\ b\texttt{;}\,(S_2\texttt{;}\,S_1)\ \texttt{endl}$$

equivalent in the Natural Semantics? If so, give a proof; if no, explain your answer.

c. What does it mean for the Natural Semantics of Exif-Loop to be deterministic? Give a possible extension of Exif-Loop such that determinism is lost. Give the Natural Semantics for your extension. Give one well-chosen counter example that shows that indeed determinism is lost.

The three parts carry 30%, 40%, and 30% of the marks, respectively.

3) Consider the following abstract syntax:

$$\begin{array}{rcl} x & \in & \textit{Variables} \\ a & \in & \textit{Arithmetic Expressions} \\ b & \in & \textit{Boolean Expressions} \\ S & \in & \textit{Statements} \end{array}$$

$$\begin{array}{rcl} a & ::= & n \mid x \mid a_1 + a_2 \mid a_1 - a_2 \mid a_1 \times a_2 \\ b & ::= & \texttt{true} \mid \texttt{false} \mid a_1 = a_2 \mid a_1 \leq a_2 \mid \neg b \mid b_1 \;\&\; b_2 \\ S & ::= & x := a \mid S_1 \texttt{;} S_2 \mid \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2 \mid \texttt{skip} \mid \texttt{from}\ a_1\ \texttt{to}\ a_2\ \texttt{do}\ S \end{array}$$

The intention of the '**from** $a_1$ **to** $a_2$ **do** $S$' is that, first, the values of $a_1$ and $a_2$ will be checked. If $a_2$ is greater than or equal to $a_1$, then $S$ will executed. After that, the entry test will be repeated, checking now the values of $a_2$ and $a_1 + 1$. This repeats until the test fails.

a. Assume the existence of the functions $a \mapsto \mathcal{A}\,[\![a]\!]\,s$ and $b \mapsto \mathcal{B}\,[\![b]\!]\,s$ that define the semantics of arithmetic and boolean expressions. Define the Natural Semantics and the Structural Operational Semantics for **From-To-Loop**. You can restrict yourself to the rules for '**from** $a_1$ **to** $a_2$ **do** $S$'.

b. Are these two semantics *equivalent*? Motivate your answer in English (in no more than 20 words), no formal proof for either answer is required. What would be a crucial lemma to show for this result?

c. Let an abstract machine be defined by:

$$\begin{array}{rcl} \texttt{inst} & ::= & \texttt{PUSH}{-}n \mid \texttt{ADD} \mid \texttt{MULT} \mid \texttt{SUB} \mid \texttt{TRUE} \mid \texttt{FALSE} \mid \texttt{EQ} \mid \texttt{LE} \\ & & \mid \texttt{AND} \mid \texttt{NEG} \mid \texttt{FETCH}{-}x \mid \texttt{STORE}{-}x \mid \texttt{NOOP} \\ & & \mid \texttt{BRANCH}(c, c) \mid \texttt{FROMTO}(c_1, c_2, c_3) \\ c & ::= & \epsilon \mid \texttt{inst} : c \end{array}$$

Give the operational semantics of this machine (you can limit yourself to the cases '**BRANCH**', and '**FROMTO**$(c_1, c_2, c_3)$').

Assuming the definition of $\mathcal{CA}$ and $\mathcal{CB}$ that, respectively, define the translation of *Arithmetic Expressions* and *Boolean Expressions* to *Code*, give the definition of translation of programs in **From-To-Loop** to *Code*.

d. Show that the above defined translation function is correct with respect to Structural Operational Semantics. The proof will follow an inductive reasoning; you only need to show the case that deals with '**from** $a_1$ **to** $a_2$ **do** $S$' and assume all other cases to be proven correct.

The four parts carry, respectively, 20%, 20%, 20%, and 40% of the marks.

4) a. Consider the space of partial functions from *State* to *State*.

i. Give the definition of a *partial ordered* set.
ii. How is the partial order relation ' $\sqsubseteq$ ' defined on functions?
iii. Give the definition of a *least upper bound.*
iv. Give the definition of a *chain.*
v. Give the definition of a *ccpo.*
vi. When is a function *monotone*?
vii. What is a *continuous* function?

b. The abstract syntax for the language **Repeat** is given by:

$$
\begin{array}{lcl}
x & \in & \textit{Variables} \\
a & \in & \textit{Arithmetic Expressions} \\
b & \in & \textit{Boolean Expressions} \\
S & \in & \textit{Statements}
\end{array}
$$

$$
\begin{array}{lcl}
a & ::= & n \mid x \mid a_1 + a_2 \mid a_1 - a_2 \mid a_1 \times a_2 \\
b & ::= & \mathbf{true} \mid \mathbf{false} \mid a_1 = a_2 \mid a_1 \leq a_2 \mid \neg b \mid b_1 \;\&\; b_2 \\
S & ::= & x := a \mid S_1 \,;\, S_2 \mid \mathbf{if}\ b\ S \mid \mathbf{abort} \mid \mathbf{skip} \\
 & & \mid \mathbf{repeat}\ S\ \mathbf{until}\ b
\end{array}
$$

Assuming the existence of suitable functions $a \mapsto \mathcal{A}\,[\![a]\!]\,s$ and $b \mapsto \mathcal{B}\,[\![b]\!]\,s$ for the semantics of both arithmetic and boolean expressions, give the *Denotational semantics* of statements.

c. *Calculate* the denotational semantics for

$$\mathbf{repeat}\ x := x \times x\ \mathbf{until}\ x \geq 16$$

The three parts carry, respectively, 35%, 30%, and 35% of the marks.