UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2000

MEng Honours Degrees in Computing Part IV
MSc in Advanced Computing
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 C470

PROGRAM ANALYSIS

Tuesday 9 May 2000, 14:30
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions

1 Consider a simple imperative language with statements of the form:

```
S ::= x := a | skip | S1; S2 |

      if b then S1 else S2 |

      while b do S
```

In answering this question, you should adopt a suitable labelling scheme for elementary blocks.

a Available Expressions Analysis determines, for each program point, which expressions must have already been computed, and not later modified, on all execution paths to the program point. A modification of the analysis detects when an expression is available in a particular variable: a non-trivial arithmetic expression is available in a variable at a program point if it has been evaluated and assigned to the variable on all paths leading to the point and the variables in the expression have not been subsequently changed.

Write down the data flow equations and any auxiliary functions for this analysis (including their types). State any assumptions that you make about the form of programs and labelling.

b Define a transformation system which uses the modified Available Expressions Analysis from part (a) to optimise programs.

2 Consider a simple imperative language with statements of the form:

```
S ::= x := a | skip | S1; S2 |
      if b then S1 else S2 |
      while b do S
```

In answering this question, you should adopt a suitable labelling scheme for elementary blocks.

a Reaching Definitions Analysis determines, for each program point, which assignments may have been made and not overwritten, when program execution reaches that point along some path. This is normally achieved by associating sets of (variable,label)-pairs with each point. A variant of the analysis just associates sets of labels with points: the idea is that given the program, a label should suffice for finding the variables that may be assigned in some elementary block bearing that label.

Write down the data flow equations and any auxiliary functions for this analysis (including their types). State any assumptions that you make about the form of programs and labelling.

It may be appropriate to introduce a set of dummy labels, each element of which indicates that a particular variable has not yet been assigned a value.

b Write down a worklist algorithm for solving the equations generated from Reaching Definitions Analysis and give a walkthrough of the algorithm for solving the equations generated from the following program:

```
x := 10; x := x + 10; y := x * x
```

3a Consider the following simple imperative language:

```
S ::= x := a | skip | S1; S2 |
      if b then S1 else S2 |
      while b do S
a ::= x | n | a1 * a2
b ::= ...
```

In answering this part of the question, you should adopt a suitable labelling scheme for elementary blocks.

Define the notion of *Monotone Framework* and *instance* of a Monotone Framework. A parity analysis detects whether the value in a variable is odd or even or undetermined (either odd or even). Using a suitable set of abstract states, write down a parity analysis for the language.

b i) What goes wrong if we naively try to apply intra-procedural techniques to languages with procedures?

ii) Give an informal definition of *valid paths*. Why is the MVP solution not a satisfactory solution to the problems encountered in part (i).

iii) Briefly sketch how the notion of "context" may be used to provide a satisfactory solution.

(The two parts carry, respectively, 60% and 40% of the marks)

4 Consider the following simple functional programming language:

```
t ::= fn x => e | e1 e2 | x | c | e1 * e2 |

      let x = e1 in e2
```

where expressions are labelled terms.

a Write down a syntax-based specification of a 0-CFA for the language which takes into account the left-to-right evaluation order imposed by call-by-value semantics.

b. A parity analysis detects whether the value of an expression is odd or even or undetermined (either odd or even). Choose a suitable representation for this information and extend the control flow analysis of part (a) to perform this data flow analysis.