COMPUTER SCIENCE TRIPOS Part IB

Thursday 8 June 1995 1.30 to 4.30

Paper 6

Answer **five** *questions.*
No more than **two** *questions from any one section are to be answered.*
Submit the answers in five **separate** *bundles each with its own cover sheet.*
Write on **one** *side of the paper only.*

SECTION A

1 Processor Architecture

What is a *data cache* and what properties of data access does it exploit? [10 marks]

Why are caches necessary for RISC processors? [5 marks]

A translation lookaside buffer (TLB) is a specialised cache. What does it typically store? [5 marks]

2 Computer Architecture

Write short notes on the following:

(*a*) static memory bit [6 marks]

(*b*) dynamic memory bit [6 marks]

(*c*) memory chip organisation [4 marks]

(*d*) memory module organisation [4 marks]

3 Digital Communication I

What is the purpose of a signalling system in a digital telephone network? Describe the operations that the signalling system should perform. [6 marks]

Describe a possible implementation of a signalling system, including a discussion of how signalling information might be carried over the network. [8 marks]

What are the advantages/disadvantages of having a separate network for signalling traffic? [3 marks]

To what extent is a signalling system different from a general-purpose distributed computation? [3 marks]

4 Graphics

In ray tracing a large computational cost is associated with determining ray–object intersections. Explain how the use of bounding volumes and space subdivision methods may reduce this cost. [20 marks]

SECTION B

5 Programming in C and C++

Write brief notes on *four* of the following aspects of the language C++. In some of the cases it may be appropriate to compare what C++ does with the situation in other programming languages such as C, ML or Modula-3.

(*a*) Overloaded functions and operators.

(*b*) Templates as a way of achieving operations on lists where the types of the items in the lists must be kept flexible.

(*c*) Consistency checks when a program is kept in several files and these are compiled at different times.

(*d*) Inheritance and virtual functions.

(*e*) The problems of writing code that is portable from one host machine to another.

[5 marks each]

6 Compiler Construction

Compare two possible mechanisms that can be used to allow efficient execution time access to free variables of functions in a programming language that permits functions to be defined within other functions. [12 marks]

Discuss whether more efficient implementations are possible if the language permits no recursive function calls, while still allowing functions to be defined within other functions. [8 marks]

7 Prolog for Artificial Intelligence

Consider the task of normalising sum expressions. For example, the sums $(a + b) + (c + d)$ and $(a + (b + (c + d)))$ may be normalised into a standard form that is left associative: $a+b+c+d$ or equivalently $((a+b)+c)+d$. Write a Prolog procedure to define predicate `normsum` such that the goal `normsum(X,Y)` succeeds when the sum expression `X` normalises to `Y`. Procedures not using the technique of difference structures will not receive full marks. [20 marks]

8 Databases

Following a directive issued by the Inquisition, the University is to set up a database that will keep track of the way in which undergraduates are supervised.

At the end of each year this database will be consulted so that reports can be prepared to show the percentage of supervisions skipped (broken down by excuse, if any, proffered).

Students' exam grades will be correlated against their previous academic record, the number of supervisions they attended and the identity of their supervisors, so that pressure can be applied to both classes of participant.

The scheme will cover supervisions individually and in groups of two, three or four. The authorities will want to be able to determine whether group size, continuity of contact with one particular supervisor or College affiliation influence the outcome.

The same database is also to be used to help coordinate and regularise the way in which research students are used as supervisors, so they will be invited to record what subjects or year-groups they feel able to cope with and how much supervising they will undertake (which amount may vary from term to term). Note that only around 50% of all supervisions are given by research students — others are given by post-doctoral research workers, the teaching staff or people from outside the department.

Design a schema for a relational database that is to record the relevant information. You should state any assumptions that lie behind the schema design that you present.

[20 marks]

SECTION C

9 Logic and Proof

Briefly describe propositional logic, first-order logic and modal logic. Outline the syntax and semantics of each. [5 marks]

Describe the main features of the sequent calculus. (It is not necessary to give the rules for particular connectives.) [2 marks]

Explain informally why the following sequent calculus rules are sound, adding provisos if necessary:

$$\frac{A, B, \Gamma \Rightarrow \Delta}{A \wedge B, \Gamma \Rightarrow \Delta} \qquad \frac{A, \Gamma \Rightarrow \Delta}{\exists x A, \Gamma \Rightarrow \Delta}$$

[4 marks]

Prove $\exists x(A \wedge B) \Rightarrow (\exists x A) \wedge B$ using the sequent calculus. Add any necessary provisos and explain why they are necessary. [9 marks]

10 Foundations of Functional Programming

What does it mean to evaluate a λ-term to a value? Discuss, contrasting call-by-value, call-by-name and call-by-need. [5 marks]

Consider the ISWIM program

```
let suc2 n = suc (suc(n))
letrec df (n) = if iszero(n) then 0
                else suc2 (df (pre(n)))
in df(9)
```

What value does it return? [2 marks]

Translate the ISWIM program to the λ-calculus, taking as primitives *if*, Y and the basic arithmetic operation. [4 marks]

Show the first six steps of the SECD machine's execution for this λ-term. [9 marks]

11 Complexity Theory

It turns out that the following program, when run using floating-point arithmetic that remains accurate to N decimal places, will compute and print the value of π correct to almost N places. The loop (which has as its main part a step which replaces the values in `a` and `b` by their arithmetic and geometric means, respectively) will be traversed about $\log(N)$ times.

```
a := 1;
b := 1/sqrt(2);
u := 1/4;
x := 1;
pn := 4;
do { p := pn;
     y := a; a := (a+b)/2; b := sqrt(y*b);
     u := u-x*(a-y)*(a-y);
     x := 2*x;
     pn := a**2/u; } while (pn<p);
print(p);
```

You are provided with procedures that can compute Fourier Transforms and their inverses with a transform on k points (using floating-point arithmetic), taking time proportional to $k \log k$. Explain how you could implement the high-precision arithmetic needed to make this program run fast. Do not discuss how the Fourier transform will be implemented — just how it is used, and assume that the floating-point accuracy achieved by the transform will be adequate for your purposes.

[14 marks]

Overall how long (as a function of N) would you expect the complete program to take to run? [6 marks]

You do not need to understand how or why this particular calculation arrives at a value for π, or why the loop is executed only $\log(N)$ times.

12 Semantics

The language IMP$'$ comprises integer and boolean expressions and commands, defined by

$$
\begin{array}{rcll}
ie & \in & Iexp & ::= \underline{n} \mid x \mid ie_1\ \underline{iop}\ ie_2 \\
be & \in & Bexp & ::= \underline{b} \mid ie_1\ \underline{bop}\ ie_2 \\
C & \in & Com & ::= \texttt{skip} \mid x := \ ie \mid (C_1; C_2) \mid \\
 & & & \qquad \texttt{if}\ be\ \texttt{then}\ C_1\ \texttt{else}\ C_2 \mid \texttt{repeat}\ C\ \texttt{until}\ be
\end{array}
$$

where $n \in \mathbb{Z}$, $b \in \{true, false\}$, $iop \in \{+, \times, -\}$, $bop \in \{<, =\}$ and $x \in Pvar$, a set of program variables.

(a) Give an annotated evaluation semantics for IMP$'$, expressing the usual behaviour of each command and expression form, which derives statements of the forms

$$ie, S \Rightarrow_I n; R \qquad be, S \Rightarrow_B b; R \qquad C, S \Rightarrow_C S'; R, W$$

for $S, S' \in States = (Pvar \to \mathbb{Z})$ and $R, W \subseteq Pvar$. $C, S \Rightarrow_C S'; R, W$ means 'in state S, command C executes to state S' whilst reading the set of variables R and writing the set of variables W'. Similarly, if $e \in Iexp \cup Bexp$ then $e, S \Rightarrow v; R$ means 'in state S, e reads variables R in evaluating to v'.

[5 marks]

(b) For $be \in Bexp$, $ie \in Iexp$, $C \in Com$ use induction on the structure of phrases to give simple definitions of sets

$$\mathcal{R}(ie), \mathcal{R}(be), \mathcal{PR}(C), \mathcal{PW}(C) \ \subseteq \ Pvar$$

where $\mathcal{R}(e)$ is the set of variables accessed by e, $\mathcal{PR}(C)$ is a set of variables possibly read during the execution of C and $\mathcal{PW}(C)$ is a set of variables possibly written to during the execution of C. Give an example to show that it is *not* in general true that

$$C, S \Rightarrow_C S'; R, W \quad \text{implies} \quad W = \mathcal{PW}(C).$$

[5 marks]

(c) Prove that for any C, S, S', R, W

$$C, S \Rightarrow_C S'; R, W \quad \text{implies} \quad (\forall x \in Pvar.\ x \notin W \Rightarrow S(x) = S'(x))$$

and that for any be, S, S', b, R

$$(\forall x \in \mathcal{R}(be). S(x) = S'(x)) \quad \text{implies} \quad (be, S \Rightarrow_B b; R \iff be, S' \Rightarrow_B b; R)$$

[5 marks]

(d) Prove that for any C_1, C_2, C_3 and be that if $\mathcal{R}(be) \cap \mathcal{PW}(C_1) = \emptyset$ then

$$(C_1; \texttt{if}\ be\ \texttt{then}\ C_2\ \texttt{else}\ C_3) \ \approx \ \texttt{if}\ be\ \texttt{then}\ (C_1; C_2)\ \texttt{else}\ (C_1; C_3)$$

You may assume without proof that if $C, S \Rightarrow_C S'; R, W$ then $W \subseteq \mathcal{PW}(C)$.

[5 marks]