COMPUTER SCIENCE TRIPOS Part Ib

Wednesday 2 June 2004 1.30 to 4.30

Paper 5

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. On each cover sheet, write the numbers of* **all** *attempted questions, and circle the number of the question attached.*

You may not start to read the questions printed on the subsequent pages of this question paper until instructed that you may do so by the Invigilator

SECTION A

1 Data Structures and Algorithms

It is proposed to store a large number of records on a disk using Larsen's method so that any lookup can be done using only one disk transfer. All the records are of length 200 bytes and each contains a 20 byte key. The data is to be held on a single disk preformatted to contain 100,000,000 sectors each of size 4096 bytes. Reading multiple consecutive sectors is regarded as a single transfer.

(a) Describe Larsen's algorithm in detail and, for the records and disk specified above, state the disk block size, the signature size and the amount of main memory that you would choose to use. [10 marks]

(b) Carefully estimate the maximum number of records that could reasonably be stored on the disk assuming the sizes you gave in part (a). [6 marks]

(c) Discuss the advantages and disadvantages of different signature sizes. [4 marks]

2 Computer Design

It is possible to design a single instruction computer (SIC). For example, the instruction Subtract and Branch on Negative is sufficiently powerful. This instruction takes the form "A,B,C,D", meaning "Read A, Subtract B, Store in C, and Branch to D if negative". If a branch is not required, the address D can be set to the next instruction in the sequence so that the next instruction will be executed regardless of whether the branch is taken or not. An assembler short form for this branchless instruction is simply "A,B,C".

(*a*) Write fully commented SIC assembler which implements the following pseudo code:

```
a=1;
b=1;
for(i=1; i<n; i++) {
  a=a+b;
  b=a-b;
}
```

[9 marks]

(*b*) Reduced instruction set computers typically achieve high performance by optimising the common case. In particular, a regular and simple instruction format typically allows extensive use of pipelining. What pipeline and memory structure would you recommend in order to execute SIC code quickly? [9 marks]

(*c*) How does the density of SIC machine code compare with current commercial processors? [2 marks]

3 Digital Communication I

(*a*) Define the terms *flat* and *hierarchical* as applied to address spaces. [2 marks]

(*b*) Give *four* examples of address spaces and state whether they are flat or hierarchical, and why. [4 marks]

(*c*) Describe class-based addresses as used in the Internet. You need not worry about precise field sizes or class names. [4 marks]

(*d*) Describe classless addresses as used in the Internet. [3 marks]

(*e*) Why were they introduced? [2 marks]

(*f*) What information must be held in a routing table when classless addresses are used? [5 marks]

4 Concurrent Systems and Applications

(*a*) Consider a simple client-server system implemented using Java Remote Method Invocation (RMI). Describe:

(*i*) how the interface between the client and the server is defined; [4 marks]

(*ii*) how a particular instance of the server is named. [4 marks]

(*b*) One operation provided by the server is to merge the contents of two hashtables, returning the result in a new hashtable. On a centralized system the operation's signature could be defined as follows:

```
Hashtable mergeTables(Hashtable a, Hashtable b)
```

Describe in detail the semantics with which parameters are passed and results are returned when this operation is implemented over RMI. [4 marks]

(*c*) The designer of a new RMI system proposes lazily copying the contents of objects that are passed between the client and the server. For instance, a large data item passed to `mergeTables` would have to be sent only if the server actually tries to access it. It is hoped that this scheme will make distributed systems faster because it will reduce the volume of data sent over the network.

(*i*) Describe a situation in which the new system is likely to be faster than traditional RMI and a separate situation in which it is likely to be slower. [2 marks each]

(*ii*) How would this new scheme change the semantics with which parameters are passed and results are returned? [4 marks]

SECTION B

5 Computer Graphics and Image Processing

(*a*) We wish to produce two algorithms: one which draws the outline of a circle and one which draws a filled circle.

(*i*) Describe an efficient algorithm which will draw a one-pixel wide outline of a circle of integer radius, R, centred on the origin. [10 marks]

(*ii*) Describe the modifications required to your algorithm to make it draw a filled circle. [3 marks]

(*b*) Given a function `drawline(x1,y1,x2,y2)`, describe an algorithm for drawing a Bezier cubic curve to a specified level of accuracy using only straight lines. [7 marks]

6 Artificial Intelligence

(*a*) Describe the way in which a problem should be represented in order to allow its solution using a *heuristic search* technique. [5 marks]

(*b*) Define what it means for a search algorithm to be *complete*, and to be *optimal*. [2 marks]

(*c*) Define what it means for a heuristic function to be *admissible*, and to be *monotonic*. [2 marks]

(*d*) Describe the operation of the $A^\star$ heuristic search algorithm. [5 marks]

(*e*) Prove that the $A^\star$ heuristic search algorithm is optimal when applied in conjunction with a monotonic heuristic. State the conditions under which the algorithm is also complete, and explain why this is the case. [6 marks]

7 Compiler Construction

You have been provided with the description of a programming language, J, intended for scripting applications. Its syntax is similar to a cut-down version of Java in that it consists of function definitions which have bodies containing if-then-else, while-do, assignments and (typed) declarations of variables. Only one statement or keyword may occur on a line so that it is sufficient to describe the start of a loop iteration with its line number. You need to explain to your boss the alternatives for implementing this so that a decision may be made as to the best implementation strategy.

The choice is between:

(a) compiling J to machine code;

(b) compiling J to "interpreted byte code", and then interpreting this;

(c) parsing J to a syntax tree representation and then interpreting this using a function which walks the tree;

(d) keeping J in a text file and then interpreting it by reading each line (and acting on it) as and when the line is required.

For *each* of (a)–(d), (i) summarise the main phases of work that are done *before* execution in each case, giving a brief explanation of the main actions of the main interpreter loop (if any) *during* execution, and (ii) for each of the following possible erroneous forms, explain whether the error would be found before or during execution: malformed syntax, undeclared variable, type error, division by zero.

[5 marks each]

You are not expected to argue for or against any of the alternatives.

8 Databases

Assume a simple movie database with the following schema. (You may assume that producers have a unique certification number, `Cert`, that is also recorded in the `Movie` relation as attribute `prodC#`; and no two movies are produced with the same title.)

```
Movie(title,year,length,prodC#)
StarsIn(movieTitle,movieYear,starName)
Producer(name,address,cert)
MovieStar(name,gender,birthdate)
```

(a) Write the following queries in SQL:

(i) Who were the male stars in the film *The Red Squirrel*? [1 mark]

(ii) Which movies are longer than *Titanic*? [2 marks]

(b) SQL has a boolean-valued operator `IN` such that the expression `s IN R` is true when `s` is contained in the relation `R` (assume for simplicity that `R` is a single attribute relation and hence `s` is a simple atomic value).

Consider the following nested SQL query that uses the `IN` operator:

```
SELECT name
FROM   Producer
WHERE  cert IN (SELECT prodC#
                FROM   Movie
                WHERE  title IN (SELECT movieTitle
                                 FROM   StarsIn
                                 WHERE  starName='Nancho Novo'));
```

(i) State concisely what this query is intended to mean. [1 mark]

(ii) Express this nested query as a single `SELECT-FROM-WHERE` query. [2 marks]

(iii) Is your query from part (b)(ii) always equivalent to the original query? If yes, then justify your answer; if not, then explain the difference and show how they could be made equivalent. [6 marks]

(c) SQL has a boolean-valued operator `EXISTS` such that `EXISTS R` is true if and only if `R` is *not* empty.

Show how `EXISTS` is, in fact, redundant by giving a simple SQL expression that is equivalent to `EXISTS R` but does not involve `EXISTS` or any cardinality operators, e.g. `COUNT`. [Hint: You may use the `IN` operator.] [8 marks]

SECTION C

9 Logic and Proof

In this question x, y, z are variables, and a, b, c are constants.

(a) Briefly outline the semantics of first order logic. [5 marks]

(b) Use the semantics of first order logic to justify that the set of formulae

$$\{\forall x(x = c),\ P(a),\ \neg P(b)\}$$

is unsatisfiable. [2 marks]

(c) For each of the following first order logic formulae: **either** prove it to be valid using the sequent calculus; **or** give an interpretation that makes it false.

$$[\forall x(\exists y(R(x, y)))] \rightarrow \exists x(R(x, x))$$
$$[\exists x(\neg P(x))] \rightarrow \neg\exists x(P(x))$$
$$[\neg\exists x(P(x))] \rightarrow \exists x(\neg P(x))$$
$$\exists x(P(x) \rightarrow P(a) \wedge P(b))$$

[2 marks each]

(d) Consider the following set Γ of first order logic formulae:

$$\left\{ \begin{array}{l} \forall x(\neg R(x, x)), \quad \forall xyz(R(x, y) \wedge R(y, z) \rightarrow R(x, z)), \\ R(a, b), \quad \forall xy(R(x, y) \rightarrow \exists z(R(x, z) \wedge R(z, y))) \end{array} \right\}$$

(i) Find an interpretation that satisfies Γ. [3 marks]

(ii) Can Γ be satisfied by an interpretation with a finite domain? Briefly justify your answer. [2 marks]

10 Foundations of Functional Programming

Pure lambda-calculus does not have any built-in control structures or arithmetic. It just has lambda-expressions.

(a) Using an untyped pure lambda calculus, explain how to model the natural numbers $0, 1, \ldots$ together with a test for zero and addition and multiplication operations. [7 marks]

(b) How can recursive function definitions be expressed in pure lambda calculus? Give a lambda-term for any special things that you need to introduce. [4 marks]

(c) Suppose you are provided with a lambda expression that works with numbers in the sense of part (a) above and which will find the predecessor of any non-zero number. Express the factorial function in terms of the primitives that you introduced in parts (a) and (b). [5 marks]

(d) Explain how a typical polymorphic typing system would respond to the various lambda-expressions you have given. If there were any that would fail to type-check, explain why: if everything you have written could be given valid polymorphic types, show what those types would be for a couple of the key expressions. [4 marks]

Credit will be given for clarity and conciseness of presentation, and justification of why your lambda expressions apply in the general case will be preferred to sets of examples that merely illustrate them in particular cases.

11 Semantics of Programming Languages

The language L has expression syntax

$$e ::= n \mid \textsf{skip} \mid \textsf{fn}\ x{:}T \Rightarrow e \mid e_1\ e_2 \mid x \mid e_1 := e_2 \mid !e \mid \textsf{ref}e \mid \ell$$

with types

$$T ::= \textsf{int} \mid \textsf{unit} \mid T_1 \rightarrow T_2 \mid T\,\textsf{ref}$$

It is intended to have a call-by-value semantics.

(*a*) Define the set of *values* for L. [2 marks]

(*b*) Give type rules and reduction rules for the store-related expressions $e_1 := e_2 \mid !e \mid \textsf{ref}e \mid \ell$. Define clearly what the type environments and stores you are using are. [10 marks]

(*c*) Discuss possible alternative choices for the semantics of the operations in part (*b*), paying particular attention to: (*i*) what can be stored, (*ii*) store initialisation, and (*iii*) the results of assignments. Illustrate your answer with type rules and reduction rules as appropriate, and comment on any pragmatic advantages or disadvantages. [8 marks]

12 Complexity Theory

Recall that a *simple path* in a graph is a path with no repeated nodes. Consider the following two decision problems:

- Given a graph $G = (V, E)$, a positive integer k, source $s \in V$ and a target $t \in V$, is there a simple path from s to t of length *at least* k?
- Given a graph $G = (V, E)$, a positive integer k, source $s \in V$ and a target $t \in V$, is there a simple path from s to t of length *at most* k?

One of these problems is known to be in P while the other one is known to be NP-complete.

(*a*) Which of the two problems is in P and which is NP-complete? [2 marks]

(*b*) Describe a polynomial time algorithm for the problem that is in P. [6 marks]

(*c*) Give a proof of NP-completeness for the problem that is NP-complete. You may assume the NP-completeness of any problem, such as *Hamiltonian Cycle*, mentioned in the lecture course. [12 marks]

END OF PAPER