COMPUTER SCIENCE TRIPOS Part II (General)
DIPLOMA IN COMPUTER SCIENCE

Thursday 5 June 2003 1.30 to 4.30

Paper 13 (Paper 4 of Diploma in Computer Science)

Answer **five** *questions.*

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

1 Data Structures and Algorithms

(*a*) In the Burrows–Wheeler Block Compression algorithm it is necessary to sort all the suffixes of a vector of possibly tens of millions of bytes.

(*i*) Explain why Shell sort using a simple character by character string comparison function is unlikely to be satisfactory. [2 marks]

(*ii*) Describe in detail the data structures and algorithms you would use to sort the suffixes and explain why your method is an improvement. You may assume that you have plenty of RAM available. [8 marks]

(*b*) In the Burrows–Wheeler Block Compression algorithm a sequence of small positive integers are transmitted using Huffman encoding.

(*i*) Describe how the Huffman code is constructed. [4 marks]

(*ii*) Outline how it can be represented compactly in the compressed file. [6 marks]

2 Computer Design

(*a*) Briefly describe the differences between a microprocessor's interface bus, a system I/O bus (such as PCI), and a peripheral interface. Consider the bandwidth characteristics and physical implementation of each. [8 marks]

(*b*) Show how four 64Mbit byte-wide DRAM chips could be interfaced to a simple 32-bit microprocessor. Give a schematic diagram showing the connections, and a timing diagram to demonstrate the operation of the control signals. [8 marks]

(*c*) Why might accesses to sequential DRAM addresses be treated differently from non-sequential ones? [4 marks]

3 Digital Communication I

(*a*) Define the term *multiplexing* as applied to communication systems. [4 marks]

(*b*) Describe *three* types of multiplexing, identifying in each case

(*i*) mechanisms by which symbols are associated with particular channels;

(*ii*) mechanisms by which transmitters are assigned channel resource;

(*iii*) characteristics of the multiplexed channels;

(*iv*) applications which are suited to the type of multiplexing. [16 marks]

4 Distributed Systems

(*a*) Discuss alternative approaches to creating unique names for objects in large-scale distributed systems or applications. [3 marks]

(*b*) Discuss the support for naming and name resolution in the following:

(*i*) the Internet; [5 marks]

(*ii*) object-oriented middleware; [5 marks]

(*iii*) message-oriented middleware; [5 marks]

(*iv*) web-based systems. [2 marks]

5 Computer Graphics and Image Processing

(*a*) We use homogeneous coordinates to represent transformations in 3D space:

$$\begin{bmatrix} x'_H \\ y'_H \\ z'_H \\ w'_H \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ a_{31} & a_{32} & a_{33} & b_3 \\ c_1 & c_2 & c_3 & d \end{bmatrix} \begin{bmatrix} x_H \\ y_H \\ z_H \\ w_H \end{bmatrix}$$

(*i*) Explain how to convert standard 3D coordinates, (x, y, z), to homogeneous coordinates and how to convert homogeneous coordinates to standard 3D coordinates. [2 marks]

(*ii*) Describe the types of transformations provided by each of the four blocks of coefficients in the matrix ($a_{11} \cdots a_{33}$, $b_1 \cdots b_3$, $c_1 \cdots c_3$, and d). [6 marks]

(*iii*) Explain what transformation is produced by each of the following matrices:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & p & -p(1+r) \\ 0 & 1 & q & -q(1+r) \\ 0 & 0 & 1+r & -r(1+r) \\ 0 & 0 & 1 & -r \end{bmatrix}$$

[4 marks]

(*b*) Describe an algorithm (in 2D) which clips an arbitrary polygon against an arbitrary axis-aligned rectangle. [8 marks]

6 Compiler Construction

Each of the following statements may be true, false, or nonsensical. Indicate which and (respectively) provide a (one-sentence) justification of why it holds, a counter-example or other explanation of why it fails, or corrected statement.

(*a*) In a language with stack-allocated free variables, the static chain pointer always points to the caller's stack frame.

(*b*) If each procedure sets up one exception handler whose scope includes all of the procedure calls it makes, then the entries in the exception handler stack and those in the dynamic chain will be in 1–1 correspondence.

(*c*) Fortran gives semantic meaning to parentheses, i.e. `a+b+c` permits compiler re-arrangement whereas `(a+b)+c` does not. This matters for floating point. However, since Fortran '`+`' is left associative the above two expressions yield the same parse tree, and we have a contradiction.

(*d*) Two alternatives to conservative garbage collection are liberal garbage collection or new labour style de-allocation.

(*e*) Any type 3 (regular) grammar is also a type 2 (context-free) grammar. Hence any lexer generated by `lex` (or `JLex`) could instead be generated by `yacc` (or `Cup`).

(*f*) When a compiler for a language like Java compiles e_1+e_2, it computes the types of e_1 and e_2 so that it can treat it as e_1+`(float)`e_2 if e_1 is of type `float` and e_2 is of type `int`.

(*g*) A syntax-tree interpreter can fail (give an error) when evaluating a variable name which does not appear in the current environment. Any program which fails using static scoping will fail using dynamic scoping.

(*h*) A syntax-tree interpreter can fail (give an error) when evaluating a variable name which does not appear in the current environment. Any program which fails using dynamic scoping will fail using static scoping.

(*i*) A dynamically-typed language is one in which the type of a value is carried around at run-time; a type error is given at run-time if values are used inappropriately. Such languages must be dynamically linked otherwise type errors will occur.

(*j*) Java `.class` files and ELF-style `.o` (or `.obj`) object files represent similar information, i.e. compiled code, a list of symbols made available to other functions, and a list of undefined symbols which the file expects to be defined elsewhere.

[2 marks each]

7 Artificial Intelligence I

A simple game works as follows. We have a board divided into n by m square cells. We also have an unlimited number of L-shaped tiles, each made to cover exactly three squares. The tiles can appear in any of the four possible orientations. Our aim is to cover the board completely with non-overlapping tiles.

(*a*) A single tile on the board can be described using a list such as `[[1,1],[1,2],[2,1]]` containing three tuples, specifying the position of each part of the tile on the board. Consider the following Prolog predicate, which is true if the six variables describe a correct, L-shaped tile.

```
tile([[A,B],[C,D],[E,F]]) :- C is A+1, D is B, E is A, F is B-1;
                             C is A+1, D is B, E is A, F is B+1;
                             C is A-1, D is B, E is A, F is B+1;
                             C is A-1, D is B, E is A, F is B-1.
```

Explain what happens in response to a query of the form

```
tile([[4,5],[B,C],[D,E]]).
```

Keep in mind the effects of backtracking. [2 marks]

(*b*) Write a Prolog predicate `goodplace([[A,B],[C,D],[E,F]],[N,M])` that is true if `[[A,B],[C,D],[E,F]]` is a validly shaped tile and all of its parts lie within an `N` by `M` board. Your predicate should behave under backtracking in such a way that the response to a query of the form

```
goodplace([[10,4],[B,C],[D,E]],[10,10]).
```

is to find the unspecified values for all tiles which have a valid shape and fall within the board. In this example there would be two such tiles. [6 marks]

(*c*) Write a Prolog predicate `tiling(Available,Solution,Size)`. Here, `Size` is the size of the board represented as above, `Solution` is a list of tiles that solves the problem, and `Available` is a list of available positions on a board of the given size. For example, if `Size` is `[2,2]` then `Available` is `[[1,1],[1,2],[2,1],[2,2]]`.

Your predicate should be true if the `Solution` given is a valid one, and should be capable of finding a valid `Solution` in response to a query such as

```
tiling([[1,1],[1,2],...,[10,10]],X,[10,10]).
```

Full marks will only be given for predicates that can exploit backtracking to find all possible solutions. [12 marks]

8 Databases

(*a*) Define the ACID properties of a transaction. [6 marks]

(*b*) Define what is meant by *strict two-phase locking* (strict 2PL). [4 marks]

(*c*) Assume that in addition to traditional Read and Write actions a DBMS supports increment and decrement actions: Inc and Dec (both are assumed to perform blind writes).

Consider the following two transactions.

$$T1 : [\mathsf{Inc}(A), \mathsf{Dec}(B), \mathsf{Read}(C)]$$
$$T2 : [\mathsf{Inc}(B), \mathsf{Dec}(A), \mathsf{Read}(C)]$$

(*i*) By considering some possible schedules of the above transactions, describe carefully the concurrency permitted by strict 2PL using just shared/read and exclusive/write locks. [3 marks]

(*ii*) Detail a way to gain more interleaving whilst still maintaining strict 2PL. [7 marks]

9 Numerical Analysis II

(*a*) Explain the term *positive semi-definite matrix.* [1 mark]

(*b*) Let $\mathbf{A}$ and $\mathbf{B}$ be $n \times n$ matrices and let $\mathbf{x}$ be a vector of n elements. State *Schwarz's inequality* for each of the products $\mathbf{AB}$ and $\mathbf{Ax}$. What are the *singular values* of $\mathbf{A}$, and how are they related to the ℓ_2 norm of $\mathbf{A}$? [4 marks]

(*c*) Describe briefly the *singular value decomposition* $\mathbf{A} = \mathbf{UWV}^T$, and how it may be used to solve the linear equations $\mathbf{Ax} = \mathbf{b}$. [4 marks]

(*d*) Let $\hat{\mathbf{x}}$ be an approximate solution of $\mathbf{Ax} = \mathbf{b}$ and write $\mathbf{r} = \mathbf{b} - \mathbf{A}\hat{\mathbf{x}}$, $\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}}$. Find an expression for an upper bound on the relative error $\|\mathbf{e}\| \;/\; \|\mathbf{x}\|$ in terms of computable quantities. Show how this formula may be computed using the singular values of $\mathbf{A}$. [8 marks]

(*e*) Suppose $\mathbf{A}$ is a 5×5 matrix and its singular values are 10^3, 1, 10^{-14}, 10^{-18}, 10^{-30}. If *machine epsilon* $\simeq 10^{-15}$ then choose a suitable *rank* for an approximate solution and form the generalised inverse $\mathbf{W}^+$. [3 marks]

10 Introduction to Functional Programming

(*a*) Explain the difference between lazy and eager evaluation, illustrating with an example. [5 marks]

(*b*) Given the following definitions

```
fun I x = x;

fun K x y = x;

fun S x y z = x z (y z);
```

give the most general type of each of the following five expressions:

(*i*) `K`

(*ii*) `K I`

(*iii*) `S`

(*iv*) `S (K S)`

(*v*) `S S (K I)`

[15 marks]

11 Natural Language Processing

(*a*) Define the following terms, as they are used in grammar formalisms for natural language:

(*i*) feature structure

(*ii*) feature structure subsumption

(*iii*) feature structure unification

[8 marks]

(*b*) Discuss the advantages and disadvantages for NLP applications of grammar formalisms that use feature structures compared with context free grammars. [12 marks]

12 Complexity Theory

(*a*) For each k, the k-clique problem is defined as the following decision problem:

> Given a graph G, does it contain a clique with at least k vertices?

Show that k-clique is in P for each k. [6 marks]

(*b*) The problem Clique is defined as the following decision problem:

> Given a graph G and an integer k, does G contain a clique with at least k vertices?

Show that Clique is NP-complete, using the assumption that 3-SAT is NP-complete. [10 marks]

(*c*) Explain why, if P=NP then there is a polynomial time algorithm for factorising numbers. [4 marks]

END OF PAPER