MATHEMATICAL TRIPOS Part IA

Monday 5 June 2006 1.30 to 4.30

PAPER 5

Before you begin, read these instructions carefully.

Answer ***two*** *questions from Section A, and* ***one*** *question from* ***each*** *of Sections B, C, D and E.*

Write on ***one*** *side of the paper only and begin each answer on a separate sheet.*

Write legibly; otherwise you place yourself at a grave disadvantage.

At the end of the examination:

Tie up your answers in ***six 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.*

STATIONERY REQUIREMENTS
Script paper
Blue cover sheets
Tags

SPECIAL REQUIREMENTS
None

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 Foundations of Computer Science

Give an example of an ML function belonging to *each* of the following complexity classes:

(*a*) $O(1)$;

(*b*) $O(n)$;

(*c*) $O(n \log n)$;

(*d*) $O(n^2)$;

(*e*) $O(2^n)$.

Each answer may contain code fragments (involving well-known functions) rather than self-contained programs, but must include justification. (The upper bound in each case should be reasonably tight.) [2 marks each]

2 Operating Systems

For *each* of the following, indicate whether the statement is true or false, and explain why this is the case (no marks will be awarded for an answer with no explanation).

(*a*) The Windows XP Executive is mostly implemented in user mode.

(*b*) Floating-point hardware can be used to invert the access matrix.

(*c*) Polled-mode I/O is sometimes preferable to interrupt-driven I/O.

(*d*) Microkernel operating systems are faster than monolithic systems.

(*e*) Windows XP is architecturally more secure than Unix.

[2 marks each]

3 Programming in Java

Suppose that you are forbidden from using `printf` or `Integer.toHexString`, or indeed any other existing library way of doing it, but you still need to display a Java integer in hexadecimal as one to eight digits. For instance, you are to display the number 19 (decimal) as the string "`13`", and -1 must come out as "`ffffffff`".

Write a Java method called `toHex` that takes an integer as its argument and returns the string form of the hexadecimal representation of that number. Explain clearly how your code works, commenting on how it avoids displaying unnecessary leading zeros and how negative numbers are handled. [10 marks]

4 Numbers and Sets

Explain what is meant by a prime number.

By considering numbers of the form $6p_1p_2\cdots p_n - 1$, show that there are infinitely many prime numbers of the form $6k-1$.

By considering numbers of the form $(2p_1p_2\cdots p_n)^2+3$, show that there are infinitely many prime numbers of the form $6k+1$. [*You may assume the result that, for a prime* $p>3$, *the congruence* $x^2 \equiv -3 \pmod{p}$ *is soluble only if* $p \equiv 1 \pmod{6}$.]

[10 marks]

SECTION B

5 Foundations of Computer Science

(*a*) This question concerns the data structure of queues.

(*i*) Describe the primitive queue operations. [3 marks]

(*ii*) Describe an efficient implementation of queues, presenting code fragments as appropriate (a complete program listing is not required). [3 marks]

(*iii*) Carefully discuss the efficiency of your implementation, using the concept of amortised time. [4 marks]

(*b*) Write an ML function to compute all permutations of its argument, a list. (You may assume that the elements of this list are distinct.) For example, given the argument $[1,2,3]$, the result should be a list consisting of the elements $[1,2,3]$, $[2,1,3]$, $[2,3,1]$, $[1,3,2]$, $[3,1,2]$ and $[3,2,1]$ in any order. For full credit, your code must be well structured and clearly explained. [10 marks]

6 Foundations of Computer Science

(*a*) Contrast *ordinary lists*, *lazy lists* and *mutable lists* by

(*i*) presenting the ML datatype declaration of each type of list, and [3 marks]

(*ii*) implementing a filter functional for each type of list. [7 marks]

(The mutable version should remove the elements that do not satisfy the given predicate, rather than constructing a new list.)

(*b*) The *intersection* of two dictionaries is the largest dictionary that agrees with them both. For example, if one dictionary is cat=3, dog=2, rabbit=9 while the other is cat=4, dog=2, hamster=9, then their intersection is dog=2.

Code an ML function to compute the intersection of two dictionaries, where dictionaries are represented by binary search trees. You may assume that the dictionary lookup and update operations are provided. For full credit, your solution must be simple and clear. [10 marks]

SECTION C

7 Operating Systems

(*a*) FIFO, LRU, and CLOCK are three page replacement algorithms.

(*i*) Briefly describe the operation of each algorithm. [6 marks]

(*ii*) The CLOCK strategy assumes some hardware support. What could you do to allow the use of CLOCK if this hardware support were not present? [2 marks]

(*iii*) Assuming good temporal locality of reference, which of the above three algorithms would you choose to use within an operating system? Why would you not use the other schemes? [2 marks]

(*b*) What is a *buffer cache*? Explain why one is used, and how it works. [6 marks]

(*c*) Which buffer-cache replacement strategy would you choose to use within an operating system? Justify your answer. [4 marks]

8 Operating Systems

(*a*) Devices are ultimately connected to the CPU via a *bus*.

(*i*) What are the main components of a bus? [3 marks]

(*ii*) Describe how the CPU uses a bus to communicate with a device. [3 marks]

(*iii*) How does the situation become more difficult when we have DMA-capable devices? [2 marks]

(*iv*) Why does a typical computer have more than one bus? [2 marks]

(*b*) A programmer at MegaCorp is given the task of optimising a program for which no source code exists; all that is available is an executable file.

(*i*) How could the programmer modify the operating system to work out which parts of the program are executed frequently, and thus might be candidates for optimisation? [4 marks]

(*ii*) The programmer determines that the slowest parts of the program involve loops which perform repeated integer multiplications, and where the multiplicand is always either 8 or 15. How could the program be modified to use faster ALU operations instead? [6 marks]

SECTION D

9 Programming in Java

For *each* of the following Java language features, give an example of a place in a large Java library where you might expect it to be used. Explain exactly what the feature achieves, why that is a benefit in the context you describe, and what risk or inconvenience might arise if the feature were not deployed.

(*a*) Methods that have been declared as `protected`. [4 marks]

(*b*) Classes that are labelled as `final`. [4 marks]

(*c*) Generic methods – that is, ones where the types of their arguments and results involve other types enclosed in angle brackets, as in `ClassName<AnotherClassName>`. [4 marks]

(*d*) Fields within a class that are marked as `private`. [4 marks]

(*e*) Parts of the library defined as an `interface` rather than as a `class`. [4 marks]

In some cases you may refer to a concrete example present in the existing Java libraries. You may also propose uses that current libraries do not exploit. Marking will pay attention to the extent to which you cover the topics you have been explicitly asked to explain: ill-structured general discussions of the Java features concerned will attract little credit.

10 Programming in Java

The "Game of Life" is played on a board of square cells. Each cell is either "live" or "dead". Initially most cells are dead, but a seeding pattern of live ones is set up. Each square cell has eight immediate neighbours (North, South, East, West, and four diagonal ones). At each time step all cells transform simultaneously. If a cell is dead, it becomes alive if it had (just before this time step) exactly three live neighbours. If it is alive, it becomes dead unless it has two or three live neighbours. In this question, locations beyond a 1000×1000 board are to be treated as permanently dead.

(*a*) Show how to set up a simple Java 2-dimensional array of `boolean` values to represent a Life Board, with all cells initially "dead". [2 marks]

(*b*) For a location (i, j) on the board, give code that will decide whether the next state of that cell should be alive or dead. Make it clear how your code copes if the cell is at the boundary of the board. [4 marks]

(*c*) Referring to part (*b*), write code that takes one board representing the current state of the game and fills in a second board-array with the state arrived at after one time step. What would happen if instead of using two arrays you wrote the new cell state directly back, using just a single copy of the board? [3 marks]

(*d*) Re-work your solution to part (*c*) so that you can perform a time step using just one board. You may need to use a 1000-element vector to store information in a way that makes the update safe. [7 marks]

(*e*) All the code you have written so far uses an array of `boolean` values. Some programmers would instead use an array of `int` values and treat each of the 32 bits in each `int` as giving the status of a cell. Suppose you have a 2-dimensional array of integers of size 1024 by 32 (that size is chosen so the array of integers may be viewed as a 1024 by 1024 array of bits): give code to retrieve a bit from a given position (i, j). [4 marks]

SECTION E

11 Numbers and Sets

Explain what is meant by an *equivalence relation* on a set A.

If R and S are two equivalence relations on the same set A, we define

$$R \circ S = \{(x,z) \in A \times A : \text{there exists } y \in A \text{ such that } (x,y) \in R \text{ and } (y,z) \in S\}.$$

Show that the following conditions are equivalent:

(i) $R \circ S$ is a symmetric relation on A.

(ii) $R \circ S$ is a transitive relation on A.

(iii) $S \circ R \subseteq R \circ S$.

(iv) $R \circ S$ is the unique smallest equivalence relation on A containing both R and S.

Show also that these conditions hold if $A = \mathbb{Z}$ and R and S are the relations of congruence modulo m and modulo n, for some positive integers m and n.

[20 marks]

12 Numbers and Sets

State and prove the Inclusion–Exclusion Principle.

A permutation σ of $\{1,2,\ldots,n\}$ is called a *derangement* if $\sigma(j) \neq j$ for every $j \leq n$. Use the Inclusion–Exclusion Principle to find a formula for the number $f(n)$ of derangements of $\{1,2,\ldots,n\}$. Show also that $f(n)/n!$ converges to $1/e$ as $n \to \infty$.

[20 marks]

END OF PAPER