COMPUTER SCIENCE TRIPOS Part IA

NATURAL SCIENCES TRIPOS Part IA (Paper CS/1)

Monday 3 June 2019 1.30 to 4.30

COMPUTER SCIENCE Paper 1

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

*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

STATIONERY REQUIREMENTS
Script paper
Blue cover sheets
Tags

SPECIAL REQUIREMENTS
Approved calculator permitted

SECTION A

1 Foundations of Computer Science

Three alternative representations for non-negative integers, n, are:

- **Peano**: values have the form `S(... S(Z) ...)`, applying `S` n times to `Z` where `S` and `Z` are constructors or constants of some data type.
- **Binary**: values are of type `bool list` with 0 being represented as the empty list, and the least-significant bit being stored in the head of the list.
- **Church**: values have the form `fn f => fn x => f(... f(x) ...)`, applying `f` n times to `x`

(a) Write ML functions for *each* of these data types which take the representation of an integer n as argument and return n as an ML `int`. [6 marks]

(b) Write ML functions for *each* of these data types which take representations of integers m and n and return the representation of $m + n$. Your answers must *not* use any value or operation on type `int` or `real`. [*Hint:* you might it useful to write a function `majority: bool*bool*bool -> bool` (which returns true when two or more of its arguments are true) and to note that the ML inequality operator '`<>`' acts as exclusive-or on `bool`.] [10 marks]

(c) Letting *two* and *three* respectively be the Church representations of integers 2 and 3, indicate whether each of the following ML expressions give a Church representation of some integer and, if so what integer is represented, and if not giving a one-line reason.

(i) *two three*

(ii) *three two*

(iii) *two* $\circ$ *three*

(iv) *three* $\circ$ *two*

[4 marks]

2 Foundations of Computer Science

(a) We are interested in performing operations on nested lists of integers in ML. A nested list is a list that can contain further nested lists, or integers. For example: `[[3, 4], 5, [6, [7], 8], []]`

We will use the datatype:

```
datatype nested_list = Atom of int
                     | Nest of nested_list list;
```

Write the code that creates a value of the type `nested_list` above. [1 mark]

(b) Write the function `flatten` that flattens a nested list to return a list of integers. [3 marks]

(c) Write the function `nested_map f n` that applies a function `f` to every `Atom` in `n`. [4 marks]

(d) What is the type of `f` in Part (c)? [1 mark]

(e) Write a function `pack_as xs n` that takes a list of integers and a `nested_list`; the function should return a new `nested_list` with the same structure as `n`, with integers that correspond to the integers in list `xs`. *Note: It is acceptable for the function to fail when the number of elements differ.* Example:

```
> pack_as [1, 2, 3] (Nest [Atom 9, Nest [Atom 8, Atom 7]]);
val it = Nest [Atom 1, Nest [Atom 2, Atom 3]]: nested_list
```

[6 marks]

(f) What does the data type `nested_zlist` correspond to? [2 marks]

```
datatype nested_zlist  = ZAtom of int
                       | ZNest of (unit -> nested_zlist list);
```

(g) Write the function that converts a `nested_zlist` to a `nested_list`. [3 marks]

SECTION B

3 Object-Oriented Programming

(*a*) You are given the following implementation for an element of a list:

```
class Element {
  int item;
  Element next;

  Element(int item, Element next) {
    super();
    this.item = item;
    this.next = next;
  }

  @Override
  public String toString() {
    return item + " " + (next == null ? "" : next);
  }
}
```

(*i*) What does the statement `super()` mean? [1 mark]

(*ii*) What is the meaning of `this` in the line `this.item = item`? [1 mark]

(*iii*) What is the purpose of the annotation `@Override`? [2 marks]

(*iv*) Rewrite the class to be immutable. You may assume that there are no sub-classes of Element. [2 marks]

(*b*) Use the immutable `Element` class to provide an implementation of an immutable class `FuncList` which behaves like an `int list` in ML. Your class should include a constructor for an empty list and methods `head`, `tail` and `cons` based on the following functions in ML. Ensure that your class behaves appropriately when the list is empty. [6 marks]

```
fun head x::_ = x;
fun tail _::xs = xs;
fun cons (x,xs) = x::xs;
```

(*c*) Another developer changes your implementation to a generic class `FuncList<T>` that can hold values of any type T.

(*i*) This means that `FuncList<T>` is no longer immutable. Explain why and what could be done to remedy this. [2 marks]

(*ii*) Java prohibits covariance of generic types. Is this restriction necessary in this case? Explain why with an example. [6 marks]

4 Object-Oriented Programming

(*a*) What is an object? [2 marks]

(*b*) Give four examples of how object-oriented programming helps with the development of large software projects and explain why each one is helpful. [8 marks]

(*c*) Explain the meaning of the Open-Closed principle. [2 marks]

(*d*) Draw a UML diagram for a design satisfying the Open-Closed principle and explain why it satisfies it. [8 marks]

SECTION C

5 Numerical Analysis

(a) Let f be a single variable real function that has at least one root α, and that admits a Taylor expansion everywhere.

(i) Starting from the truncated form of the Taylor expansion of $f(x)$ about x_n, derive the recursive expression for the Newton-Raphson (NR) estimate x_n of the root at the $(n+1)$th step. [1 mark]

(ii) Consider the general Taylor expansion of $f(\alpha)$ about x_n. Using big O notation for an appropriate Taylor remainder and denoting the NR error at the nth step by e_n, prove that the NR method has quadratic convergence rate. That is, show that e_{n+1} is proportional to e_n^2 plus a bounded remainder. State the required conditions for this to hold, paying attention to the interval spanned during convergence. [6 marks]

(iii) Briefly explain two of the known problems of the NR method from an implementation standpoint or otherwise. [2 marks]

(b) Let $f(x) = x^2 - 1$. Suppose we wish to find the positive root of f using the Newton-Raphson (NR) method starting from an initial guess $x_0 \geq 1$.

(i) Show that if $x_0 \geq 1$ then $x_n \geq 1$ for all $n \geq 1$. [3 marks]

(ii) Thus find an upper bound for NR's x_{n+1} estimate in terms of x_n and in turn find an upper bound for x_n in terms of x_0. [5 marks]

(iii) Using the above, estimate the number of NR iterations to obtain the root with accuracy 10^{-9} for a wild initial guess $x_0 = 10^9$. [*Hint:* You may wish to approximate 10^3 by 2^{10}.] [3 marks]

6 Numerical Analysis

(*a*) You are given a system of real equations in matrix form $Ax = b$ where A is non-singular. Give three factorization techniques to solve this system, depending on the shape and structure of A: tall, square, symmetric. For each technique, give the relevant matrix equations to obtain the solution x, and point out the properties of the matrices involved. Highlight one potential problem from an implementation (computer representation) standpoint. [*Note:* You do not need to detail the factorization steps that give the matrix entries.] [5 marks]

(*b*) We want to estimate travel times between stops in a bus network, using ticketing data. The network is represented as a directed graph, with a vertex for each bus stop, and edges between adjacent stops along a route. For each edge $j \in \{1, \ldots, p\}$ let the travel time be d_j. The following ticketing data is available: for each trip $i \in \{1, \ldots, n\}$, we know its start time s_i, its end time f_i, and also the list of edges it traverses. The total trip duration is the sum of travel times along its edges.

We shall estimate the d_j using linear least squares estimation, i.e. solve $\arg\min_\beta \|y - X\beta\|^2$ for a suitable matrix X and vectors β and y.

(*i*) Give an example of ticket data for a trip traversing 5 edges, and write the corresponding equation of its residual. [1 mark]

(*ii*) Give the dimensions and contents of X, β, and y for this problem. State a condition on X that ensures we can solve for β. [3 marks]

(*iii*) Give an example with $p = 2$ and $n = 3$ for which it is *not* possible to estimate the d_j. Compute $X^T X$ for your example. [2 marks]

(*c*) Let A be an $n \times n$ matrix with real entries.

(*i*) We say that A is *diagonalisable* if there exists an invertible $n \times n$ matrix P such that the matrix $D = P^{-1}AP$ is diagonal. Show that if A is diagonalisable and has only one eigenvalue then A is a constant multiple of the identity matrix. [3 marks]

(*ii*) Let A be such that when acting on vectors $x = [x_1, x_2, \ldots, x_n]^T$ it gives $Ax = [x_1, x_1 - x_2, x_2 - x_3, \ldots, x_{n-1} - x_n]^T$. Write out the contents of A and find its eigenvalues and eigenvectors. Scale the eigenvectors so they have unit length (i.e. so their magnitude is equal to 1). [6 marks]

SECTION D

7 Algorithms

(a) The Post Office of Maldonia issued a new series of stamps, whose denominations in cents are a finite set $D \subset \mathbb{N}\backslash\{0\}$, with $1 \in D$. Given an arbitrary value $n \in \mathbb{N}\backslash\{0\}$ in cents, the problem is to find a minimum-cardinality multiset of stamps from D whose denominations add up to exactly n.

In the context of solving the problem with a bottom-up dynamic programming algorithm...

(i) Give and clearly explain a formula that expresses the optimal solution in terms of optimal solutions to subproblems. [*Note:* If your formula gives only a scalar metric (e.g. the number of stamps) rather than the actual solution (e.g. which stamps), please also explain how to obtain the actual optimal solution.] [4 marks]

(ii) Draw and explain the data structure your algorithm would use to accumulate the intermediate solutions. [2 marks]

(iii) Derive the big-Theta space and time complexity of your algorithm. [1 mark]

(b) Repeat (a)(i)–(a)(iii) for the following problem:

A car must race from point A to point B along a straight path, starting with a full tank and stopping as few times as possible. A full tank lets the car travel a given distance l. There are n refuelling stations $s_o \equiv A, s_1, s_2, \ldots, s_n \equiv B$ along the way, at given distances $d_0 = 0, d_1, d_2, \ldots, d_n$ from A. The distance between adjacent stations is always less than l. The problem is to find a minimum-cardinality set of stations where the car ought to refill in order to reach B from A. [7 marks]

(c) Which of the two previous problems might be solved more efficiently with a greedy algorithm? Indicate the problem and describe the greedy algorithm. Then give a clear and rigorous proof, with a drawing if it helps clarity, that your greedy algorithm always reaches the optimal solution. Derive the big-Theta time complexity. [6 marks]

8 Algorithms

(a) Consider a Binary Search Tree. Imagine inserting the keys 0, 1, 2, ..., n (in that order) into the data structure, assumed initially empty.

(i) Draw a picture of the data structure after the insertion of keys up to $n = 9$ included. [2 marks]

(ii) Clearly explain, with a picture if helpful, how the data structure will evolve for arbitrary n, and derive the worst-case time complexity for the whole operation of inserting the $n + 1$ keys. [2 marks]

(b) Repeat (a)(i) and (a)(ii) for a 2-3-4 tree, with some scratch work showing the crucial intermediate stages. [2+2 marks]

(c) ... and for a B-tree with $t = 3$, again showing the crucial intermediate stages. [2+2 marks]

(d) ... and for a hash table of size 7 that resolves collisions by chaining. [2+2 marks]

(e) ... and for a binary min-heap. [2+2 marks]

9 Algorithms

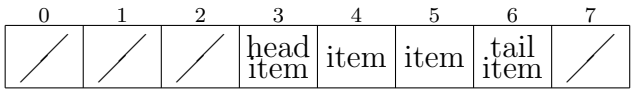
A Random Access Queue supports the operations `pushright(`x`)` to add a new item x to the tail, `popleft()` to remove the item at the head, and `element_at(`i`)` to retrieve the item at position i without removing it: $i = 0$ gives the item at the head, $i = 1$ the following element, and so on.

(a) We can implement this data structure using a simple linked list, where `element_at(`i`)` iterates from the head of the list until it reaches position i.

(i) State the complexity of each of the three operations. [1 mark]

(ii) A colleague suggests that, by defining a clever potential function, it might be possible to show that all operations have amortized cost $O(1)$. Show carefully that your colleague is mistaken. [6 marks]

(b) We can also implement this data structure using an array. The picture below shows a queue holding 4 items, stored within an array of size 8. When new items are pushed, it may be necessary to create a new array and copy the queue into it. The cost of creating an array of size n is $\Theta(n)$.



(i) Give pseudocode for the three operations. Each operation should have amortized cost $O(1)$. [6 marks]

(ii) Prove that the amortized costs of your operations are indeed $O(1)$. [7 marks]

10 Algorithms

An *undirected network* is an undirected graph in which each edge $u \leftrightarrow v$ has an associated capacity $c(u \leftrightarrow v) > 0$. Let there be a source vertex s and a sink vertex t. We wish to find the maximum flow from s to t such that on every edge $u \leftrightarrow v$ either there is flow $u \to v$, or there is flow $v \to u$, or there is no flow at all. It is not allowed to have flow in both directions simultaneously. The flow on an edge, whichever direction it is in, must not exceed capacity. We can write the flow constraints in mathematical notation as

$$\min\big\{f(u \to v),\, f(v \to u)\big\} = 0 \quad \text{and} \quad \max\big\{f(u \to v),\, f(v \to u)\big\} \leq c(u \leftrightarrow v)$$

where $f(u \to v)$ is the flow from u to v on edge $u \leftrightarrow v$. Note that this differs from the conventional Ford–Fulkerson setup, in which each *directed* edge has a capacity constraint.

(*a*) Define the *value* of a flow. State the flow conservation equation. [2 marks]

(*b*) Give an algorithm for finding a maximum flow. [6 marks]

(*c*) Prove that your algorithm returns a valid flow. [4 marks]

(*d*) Prove that your algorithm returns a maximum flow. [4 marks]

(*e*) Define the *capacity of a cut in an undirected network*. Show that the maximum flow value is equal to the minimum cut capacity. [4 marks]

[*Hint:* You may refer to code from lecture notes without repeating it in your answer. You may quote without proof any theorems from lecture notes.]

END OF PAPER