COMPUTER SCIENCE TRIPOS Part IB

Thursday 5 June 2008 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. 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
None

SECTION A

1 Economics and Law

(*a*) Show how a monopolist can increase his profits by price discrimination. [8 marks]

(*b*) Is this efficient? Justify your answer. [4 marks]

(*c*) What objections can be made to it? [4 marks]

(*d*) How might the monopolist deal with the objections? [4 marks]

2 Computer Design

(*a*) What do *RISC* and *CISC* stand for and what are the differences in practice? [6 marks]

(*b*) Some instruction sets use a register file and others use an operand stack for intermediate storage. How does the code density compare between these two approaches? [4 marks]

(*c*) Early computers used just an accumulator rather than a register file or stack. How does the code density compare between accumulator and stack machines? [4 marks]

(*d*) Compact RISC instruction sets typically use a fixed 16-bit instruction size. If three operands are to be specified, each of n bits, then $(16 - 3 \times n)$ bits are left for the opcode. If $n = 3$ then we cannot access enough registers and if $n = 4$ we do not have enough opcode bits. In practice we usually need 5 bits for the opcode leaving 11 bits to specify three registers. Using 11 bits to store three operands, how many registers can be specified and how might these three operands be decoded? [6 marks]

3 Digital Communication I

(*a*) For *each* of these examples of addressing, state whether it is flat or hierarchical and why:

(*i*) postal;

(*ii*) telephone;

(*iii*) Ethernet (MAC) address;

(*iv*) Internet (IP) address. [4 marks]

(*b*) Compare class-based and classless addresses as used in the Internet. [4 marks]

(*c*) Why were classless addresses introduced? [1 mark]

(*d*) Consider a router of IP packets.

(*i*) What information must be held in a routing-table when classless addresses are used? [3 marks]

(*ii*) Describe *Longest-Prefix Match*, providing an example of its use. [3 marks]

(*iii*) Describe the process of routing-table lookup that leads to the default-route being used and comment on the circumstance in which an IP router does not have a default-route. [3 marks]

(*e*) Considering your answers to part (*d*), describe *two* challenges for router-vendors following the introduction of classless addressing. [2 marks]

4 Concurrent Systems and Applications

(*a*) What are *Class Literals* in the context of generics in Java? [2 marks]

(*b*) Under what circumstances might *type erasure* occur in a Java program that uses generics? Illustrate your answer with sample Java code. [4 marks]

(*c*) Consider the two overloaded methods shown below. Is it valid to have these overloaded methods in a Java class? Explain your answer.

```
public static <T extends Long>
 void myMethod(List<T> x)
{
 // ...
}

public static <S extends String>
 void myMethod(List<S> x)
{
 // ...
}
```

[5 marks]

(*d*) Consider two class definitions: `A`, and `B` which extends `A`. In the context of the relationship between the types `A[]` and `B[]`, explain the difference between covariance and contravariance. Is Java covariant or contravariant? [4 marks]

(*e*) If a field is declared with `protected` access, from where can it be accessed? Is there any situation in which a `private` field defined on some class `A` can be accessed but a `protected` field defined on `A` cannot? [5 marks]

SECTION B

5 Computer Graphics and Image Processing

(*a*) Assume that you have an algorithm that can fill 3D triangles with a constant colour. Explain what additional information and additions to the algorithm are required to Gouraud shade the triangles. [6 marks]

(*b*) Given the algorithm in (*a*), explain what additional information and additions to the algorithm are required to texture map the triangles using bilinear interpolation, including an explanation of how the bilinear interpolation is done. [6 marks]

(*c*) Explain the advantages and disadvantages of using nearest-neighbour interpolation compared with bilinear interpolation. [3 marks]

(*d*) Explain why a MIPmap would be useful for texture mapping and how one could be incorporated into the algorithm from (*b*). [5 marks]

6 Compiler Construction

(*a*) Languages like Lisp, Prolog and Python are said to be *dynamically typed*. Explain this concept and its implications for the size of a run-time storage cell needed to hold a value which may be an integer or floating-point value. [4 marks]

(*b*) Consider the following object-oriented program in Java style:

```
class A { int a,b; };
class B extends A { int c,d1,d2,d3,d4,d5,d6,d7,d8,d9; };
...
static void f(A x) { x.a = 1; }
static void g(B x) { x.c = 2; }
static void h() { A p = new A(); f(p); g(p); }
static void k() { B p = new B(); f(p); g(p); }
static void main() { h(); k(); }
```

(*i*) Explain the run-time structure of values of type `A` and `B`. Indicate a constraint on the layout of these structures needed to support *inheritance*. [3 marks]

(*ii*) Indicate why the above program would not compile in Java and insert a single *cast* to make it compile. Why are two casts not required? [2 marks]

(*iii*) What happens when your Java program in part (*ii*) is executed? [2 marks]

(*iv*) Make an analogy to part (*a*) to argue why a Java value of type `A` requires more storage than that required for two integers. [2 marks]

(*v*) C++ traditionally allows values of type `A` to occupy just the space required by two integers. Comment on the implications for safety if this were allowed in Java. [2 marks]

(*c*) Explain where storage for `new` comes from. Some languages have a primitive `dispose` which de-allocates space allocated by `new`, but Java does not. Explain the implications of this for Java implementation, particularly how a program can perform `new A()` every millisecond but never run out of memory. Suppose that, while executing such a program, a `new B()` is executed. Explain, giving reasons, whether this is guaranteed to succeed in a situation where exactly half the memory available for `new` is in use. [5 marks]

7 Concepts in Programming Languages

(*a*) Explain what is understood by *static* and *dynamic* scope in the context of programming languages.

Write a program fragment in pseudocode such that its execution under static scoping and under dynamic scoping yields different outcomes. Justify your answer.

[7 marks]

(*b*) The type system of the programming language Pascal has a rich set of data-structuring concepts, including *variant records*.

Recall that variant records have a part common to all records of the type and a variable part (discriminated via an optional tag field) specific to some subset of records. For instance, consider the variant-record type `UBtree` of unary/binary branching trees below:

```
type kind = (unary,binary) ;

type UBtree = record
                value: integer ;
                case k: kind of
                  unary: ^UBtree ;
                  binary: record
                            left: ^UBtree ;
                            right: ^UBtree
                          end
              end ;
```

Explain how variant records introduced weaknesses into the Pascal type system.

Give an encoding of the `UBtree` type as a datatype in SML, and explain why the aforementioned weaknesses do not arise in SML.

[7 marks]

(*c*) Explain how *function types* are encoded in the programming language Scala, exemplifying your answer.

[6 marks]

8 Databases

(*a*) Define the notion of a *safe query* in the relational calculus. [2 marks]

(*b*) Suppose that we have schemas $R(A,\ B)$ and $S(B,\ C)$, and that the number of tuples in R is r and the number of tuples in S is s. Suppose that both R and S are not empty, and that neither contains duplicates.

For each of the following relational algebra queries, state in terms of r and s the *minimum possible* and *maximum possible* number of tuples in the result.

(*i*) $\sigma_p(R \times S)$ [2 marks]

(*ii*) $\pi_{A,\ C}(R \times S)$ [2 marks]

(*iii*) $\pi_B(R) - (\pi_B(R) - \pi_B(S))$ [2 marks]

(*iv*) $R \overset{\circ}{\bowtie}_L S$ (left outerjoin) [2 marks]

(*v*) $R \overset{\circ}{\bowtie} S$ (full outerjoin) [2 marks]

(*c*) Again, suppose that we have schemas $R(A,\ B)$ and $S(B,\ C)$. Make no assumptions about functional dependencies. Let b be some value from domain B. Consider the following relational algebra queries.

1. $\pi_{A,C}(R \bowtie \sigma_{B=b}(S))$
2. $\pi_A(\sigma_{B=b}(R)) \times \pi_C(\sigma_{B=b}(S))$
3. $\pi_{A,C}(\pi_A(R) \times \sigma_{B=b}(S))$

Two of these queries always return the same result, while one may not. Which one is different? Give a simple database instance in which this query returns a different result.

[8 marks]

SECTION C

9 Foundations of Functional Programming

(*a*) For λ-terms, define

(*i*) α-equivalence;

(*ii*) β-reduction;

(*iii*) η-reduction;

(*iv*) equality.

You may assume appropriate syntactic definitions such as FV (free variables), variable swapping, and capture-avoiding substitution. [8 marks]

(*b*) Prove or disprove the following equalities:

(*i*) $(\lambda x.\,(\lambda y.\,x\,y))\,(\lambda x.\,y) = \lambda z.\,y$ [3 marks]

(*ii*) $(\lambda x.\,M)\,((\lambda y.\,N)\,P) = (\lambda y.\,(\lambda x.\,M)N)P \quad$ where $y \notin FV(M)$.

[Hint: If $y \notin FV(M)$, then $\forall L.\,M[L/y] \equiv M$.] [4 marks]

(*c*) Show how adding the following reduction rule makes all λ-terms equal:

$$M\,N \rightarrow_s N\,M$$

[Hint: You may find it useful to recall $(\lambda x.\,M)\,L = M$ if $x \notin FV(M)$.] [5 marks]

10 Computation Theory

(*a*) Explain what is meant by each of the following statements:

(*i*) "c is a code for the total recursive function $f : \mathbb{N} \to \mathbb{N}$." [2 marks]

(*ii*) "F is a recursively enumerable set each of whose elements is a total recursive function $f : \mathbb{N} \to \mathbb{N}$." [3 marks]

(*b*) In each of the following cases state with reasons whether the set is recursively enumerable:

(*i*) the set A of all total recursive functions $a : \mathbb{N} \to \mathbb{N}$ such that $a(n+1) \geqslant a(n)$ for all $n \in \mathbb{N}$ [6 marks]

(*ii*) the set D of all total recursive functions $d : \mathbb{N} \to \mathbb{N}$ such that $d(n+1) \leqslant d(n)$ for all $n \in \mathbb{N}$ [9 marks]

11 Semantics of Programming Languages

Below is the syntax and type system for a simple functional language.

Integers $n \in \mathbb{Z} = \{..., -1, 0, 1, ...\}$
Variables $x \in \mathbb{X} = \{\mathrm{x}, \mathrm{y}, \mathrm{z}, ...\}$
Types $T ::= \mathsf{int} \mid T \to T$
Type environments Γ, finite partial functions from variables to types.
Expressions $e ::= n \mid x \mid \textbf{fn}\ x{:}T \Rightarrow e \mid e_1\ e_2$

(int) $\Gamma \vdash n{:}\mathsf{int}$ for $n \in \mathbb{Z}$ (var) $\Gamma \vdash x{:}T$ if $\Gamma(x) = T$

(fn) $$\frac{\Gamma, x{:}T \vdash e{:}T'}{\Gamma \vdash \textbf{fn}\ x{:}T \Rightarrow e : T \to T'}$$ (app) $$\frac{\Gamma \vdash e_1{:}T \to T' \qquad \Gamma \vdash e_2{:}T}{\Gamma \vdash e_1\ e_2{:}T'}$$

(*a*) Give a call-by-value operational semantics for this language, defining a judgement $e \longrightarrow e'$ (the language does not have store operations, so you can take configurations to be just expressions). [4 marks]

(*b*) Give an example of a stuck configuration. [1 mark]

(*c*) Prove the substitution lemma stated below:

If $\Gamma, x{:}T \vdash e'{:}T'$ and $\Gamma \vdash e{:}T$ with $x \notin \mathrm{dom}(\Gamma)$ then $\Gamma \vdash \{e/x\}e'{:}T'$.

[9 marks]

(*d*) State and prove type preservation, using this substitution lemma. [6 marks]

12 Complexity Theory

(*a*) Give definitions for the complexity classes SPACE(f) (for any function f); **L** and **NL**. [6 marks]

(*b*) Consider the following decision problem:

Reachability: Given a graph $G = (V, E)$ and two distinguished vertices $s, t \in V$, does G contain a path from s to t?

(*i*) Explain why **Reachability** is in the complexity class **NL**. [7 marks]

(*ii*) Show that if **Reachability** were in the class **L**, we would have **L** = **NL**. [7 marks]

END OF PAPER