COMPUTER SCIENCE TRIPOS Part IB

Thursday 5 June 2014 1.30 to 4.30 pm

COMPUTER SCIENCE Paper 6

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

STATIONERY REQUIREMENTS	SPECIAL REQUIREMENTS
Script paper	*Approved calculator permitted*
Blue cover sheets	
Tags	
Rough work pad	

1 Complexity Theory

(*a*) Give the two definitions of the complexity class NP, one using the term Turing machine and one using the term verifier. [4 marks]

(*b*) For each of the following statements, state whether it is true, false or unknown. In each case, give justification for your answer. In particular, if the truth statement is unknown, state any implications that might follow from it being true or false. [2 marks each]

(*i*) $\mathsf{3SAT} \leq_P \mathsf{CLIQUE}$

(*ii*) $\mathsf{TSP} \in \mathsf{P}$

(*iii*) $\mathsf{NL} \subseteq \mathsf{P}$

(*iv*) $\mathsf{PSPACE} \neq \mathsf{NPSPACE}$

(*c*) Let $\Sigma = \{0, 1\}$. Prove that $\emptyset$ and $\{0, 1\}^*$ are the only languages in P which are not complete for P with respect to polynomial-time reductions. [8 marks]

2 Complexity Theory

(*a*) State precisely what it means for a language (*i*) to be co-NP-complete, (*ii*) to be in NL and (*iii*) to be in PSPACE. [6 marks]

(*b*) Consider the following two decision problems.

Problem 1: Given an undirected graph $G = (V, E)$ with $|V|$ even, does G contain a clique with at least $|V|/2$ vertices?

Problem 2: Given an undirected graph $G = (V, E)$, does G contain a clique with at least $|V| - 3$ vertices?

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

(*ii*) For the problem in P, describe a polynomial-time algorithm. [4 marks]

(*iii*) For the other problem, prove that it is NP-complete. [8 marks]

3 Computation Theory

(*a*) Explain how to code register machine programs P as numbers $\ulcorner P \urcorner \in \mathbb{N}$ so that each $e \in \mathbb{N}$ can be decoded to a unique register machine program *prog*(e). [10 marks]

(*b*) Find a number $e_1 \in \mathbb{N}$ for which *prog*(e_1) is a register machine program for computing the function $\mathit{one} \in \mathbb{N} \to \mathbb{N}$ with $\mathit{one}(x) = 1$ for all $x \in \mathbb{N}$. [2 marks]

(*c*) Why is it important for the theory of computation that the functions involved in the coding and decoding given in part (*a*) are themselves register machine computable? (You are not required to prove that they are computable.) [2 marks]

(*d*) Define what it means for a set of numbers $S \subseteq \mathbb{N}$ to be register machine *decidable*. [2 marks]

(*e*) Let $\varphi_e \in \mathbb{N} \rightharpoonup \mathbb{N}$ denote the partial function of one argument computed by the register machine with program *prog*(e). Prove that $\{e \in \mathbb{N} \mid \varphi_e = \mathit{one}\}$ is register machine undecidable (where *one* is the function mentioned in part (*b*)). State carefully any standard results that you use in your proof. [4 marks]

4 Computation Theory

(*a*) Give the recursion equations for the function $\rho^n(f, g) \in \mathbb{N}^{n+1} \to \mathbb{N}$ *defined by primitive recursion* from functions $f \in \mathbb{N}^n \to \mathbb{N}$ and $g \in \mathbb{N}^{n+2} \to \mathbb{N}$. [2 marks]

(*b*) Define the class PRIM of *primitive recursive functions*, giving exact definitions for all the functions and operations you use. [5 marks]

(*c*) Show that the addition function $add(x, y) = x + y$ is in PRIM. [2 marks]

(*d*) Give an example of a function $\mathbb{N}^2 \to \mathbb{N}$ that is not in PRIM. [3 marks]

(*e*) The Fibonacci function $fib \in \mathbb{N} \to \mathbb{N}$ satisfies $fib(0) = 0$, $fib(1) = 1$ and $fib(x + 2) = fib(x) + fib(x + 1)$ for all $x \in \mathbb{N}$.

(*i*) Assuming the existence of primitive recursive functions $pair \in \mathbb{N}^2 \to \mathbb{N}$, $fst \in \mathbb{N} \to \mathbb{N}$ and $snd \in \mathbb{N} \to \mathbb{N}$ satisfying for all $x, y \in \mathbb{N}$

$$fst(pair(x, y)) = x \ \wedge \ snd(pair(x, y)) = y$$

prove by mathematical induction that any function $g \in \mathbb{N} \to \mathbb{N}$ satisfying

$$\begin{aligned} g(0) &= pair(0, 1) \\ g(x + 1) &= pair(snd(g(x)), fst(g(x)) + snd(g(x))) \end{aligned}$$

for all $x \in \mathbb{N}$, also satisfies

$$\forall x \in \mathbb{N}(fst(g(x)) = fib(x) \ \wedge \ snd(g(x)) = fib(x + 1)).$$

[4 marks]

(*ii*) Deduce that the Fibonacci function fib is in PRIM. [4 marks]

5 Logic and Proof

(*a*) Proof methods for propositional logic include the sequent calculus, DPLL and BDDs. Describe briefly each of these methods. State, with reasons, which method is to be preferred for a problem that makes heavy use of the $\leftrightarrow$ and $\oplus$ symbols. (Note that $\oplus$ denotes exclusive or.) [7 marks]

(*b*) Describe briefly the procedure for constructing a BDD, illustrating your answer using the formula $((P \lor Q) \land R) \lor (P \to (Q \land R))$.

[7 marks]

(*c*) Consider the following set of $n + 1$ propositional formulas, where $n \geq 0$:

$$\begin{aligned} &P_i \leftrightarrow P_{i+1} \qquad (\text{for } i = 1, \ldots, n) \\ &P_1 \oplus P_{n+1} \end{aligned}$$

Describe a possible execution of the DPLL procedure to determine whether this set is satisfiable or not. [6 marks]

6 Logic and Proof

(*a*) Describe briefly the concept of a decision procedure, listing at least three separate examples of decidable theories. [4 marks]

(*b*) Outline the basic ideas behind Fourier-Motzkin variable elimination, demonstrating them with reference to the following small set of constraints:

$$x + 2y \geq 10 \qquad x + z \leq 5 \qquad y \leq 3 \qquad z - 2 \geq 0$$

[6 marks]

(*c*) Call a clause *positive* if it consists of positive literals only. *Negative selection* is a refinement of resolution where two clauses can be resolved only if one of them is positive; if a clause contains any negative literals, then only one of those may be resolved with a literal in another (necessarily positive) clause. Negative selection reduces the number of combinations of literals to be compared, thereby improving performance. Consider the following set of clauses:

$$\{R(0), R(1)\} \qquad \{P(h(z)), \neg R(z)\} \qquad \{\neg P(x), \neg R(y)\}.$$

With negative selection, the first resolution step must involve $\{R(0), R(1)\}$, as no other positive clauses are available at the start.

(*i*) If a set of clauses includes no positive clauses, can it be unsatisfiable? Justify your answer. [3 marks]

(*ii*) Use resolution with negative selection to derive a contradiction from the clauses above. [7 marks]

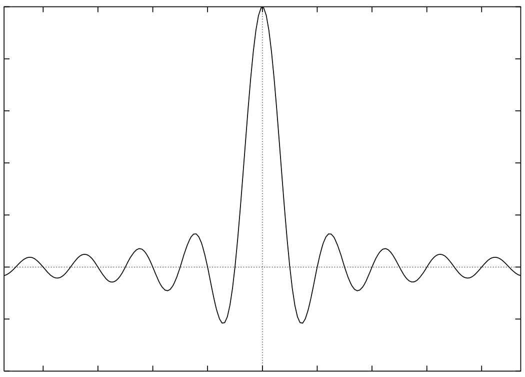
7 Mathematical Methods for Computer Science

(*a*) Using complex exponentials, prove the following trigonometric identity, which describes the multiplicative modulation of one cosine wave by another as being simply the sum of a different pair of cosine waves:

$$\cos(ax)\cos(bx) = \frac{1}{2}\cos((a+b)x) + \frac{1}{2}\cos((a-b)x)$$

[3 marks]

(*b*) The function $\text{sinc}(x) = \dfrac{\sin(\pi x)}{\pi x}$ for $x \neq 0$ as plotted here plays an important role in the Sampling Theorem. By considering its Fourier transform, show that this function is unchanged in form after convolution with itself, and show that it even remains unchanged in form after convolution with any higher frequency sinc function $\text{sinc}(ax)$ for $a > 1$, but that if $0 < a < 1$, then the result is instead that lower frequency sinc function $\text{sinc}(ax)$.



[5 marks]

(*c*) Let V be an inner product space spanned by an orthonormal system of vectors $\{e_1, e_2, \ldots, e_n\}$ so that $\forall i \neq j$ the inner product $\langle e_i, e_j \rangle = 0$, but every e_i is a unit vector so that $\langle e_i, e_i \rangle = 1$. We wish to represent a data set consisting of vectors $u \in \text{span}\{e_1, e_2, \ldots, e_n\}$ in this space as a linear combination of the orthonormal vectors: $u = \sum_{i=1}^{n} a_i e_i$. Derive how the coefficients a_i can be determined for any vector u, and comment on the computational advantage of representing the data in an orthonormal system. [7 marks]

(*d*) Show how a generating (or "mother") wavelet $\Psi(x)$ can spawn a family of "daughter" wavelets $\Psi_{jk}(x)$ by simple shifting and scaling operations, and explain the advantages of representing continuous functions in terms of such a family of self-similar dilates and translates of a mother wavelet. [5 marks]

8 Mathematical Methods for Computer Science

Suppose that X is a random variable with moment generating function $M_X(t)$ which you may assume is well-defined and finite for all t.

(*a*) Show that for any constant a and for all $t \geq 0$

$$\mathbb{P}(X \geq a) \leq e^{-ta} M_X(t) \,.$$

[5 marks]

(*b*) Show that for any constant a

$$\mathbb{P}(X \geq a) \leq e^{-f(a)}$$

where

$$f(a) = \max_{t \geq 0} \left(ta - \ln M_X(t) \right) \,.$$

[5 marks]

(*c*) Let $X_1, X_2, \ldots$ be a sequence of independent random variables each with the same distribution as X. Show that for any $a > \mathbb{E}(X)$

$$\mathbb{P}\left(\frac{1}{n} \sum_{i=1}^{n} X_i \geq a \right) \leq e^{-nf(a)} \,.$$

[5 marks]

(*d*) Show that $\mathbb{P}(X \geq a) \leq e^{-a^2/2}$ when $X \sim N(0,1)$ is a standard Normal random variable and $a > 0$. You may use the result that in this case $M_X(t) = e^{t^2/2}$.

[5 marks]

9 Semantics of Programming Languages

Consider the concurrent imperative language L with syntax and conventional operational semantics as below.

$$
\begin{aligned}
\textit{statement},\ s ::= {} & \textbf{skip} \mid x := e; s \mid \textbf{let}\, r = x\, \textbf{in}\, s \mid \textbf{let}\, r = op(e_1, ..., e_n)\, \textbf{in}\, s \\
& \mid \textbf{if}\ (e_1 = e_2)\ s\ \textbf{else}\ s'
\end{aligned}
$$

$$\textit{expression},\ e ::= \ r \mid v$$

$$\textit{process},\ p ::= \ \textit{tid}{:}s \mid p|p'$$

$$\textit{label},\ l ::= \ \mathbf{W}x{=}v \mid \mathbf{R}x{=}v \mid \tau \mid \textit{tid}{:}l \mid \mathbf{L}x$$

Here x and r range over shared and thread-local variables, op over built-in operators, v over values $0, 1, \ldots$, tid over thread ids $a, b, \ldots$. Let m range over memory states, functions from shared variables to values. In the **let**s, r binds in s.

$$\boxed{s \xrightarrow{l} s'}$$

$$\frac{}{x := v; s \xrightarrow{\mathbf{W}x=v} s}\ \text{WR} \qquad \frac{}{\textbf{let}\, r = x\, \textbf{in}\, s \xrightarrow{\mathbf{R}x=v} \{v/r\}s}\ \text{RD} \qquad \frac{}{\textbf{if}\ (v = v)\ s\ \textbf{else}\ s' \xrightarrow{\tau} s}\ \text{IF1}$$

$$\frac{v \neq v'}{\textbf{if}\ (v = v')\ s\ \textbf{else}\ s' \xrightarrow{\tau} s'}\ \text{IF2} \qquad \frac{v = [\![op]\!](v_1, \ldots, v_n)}{\textbf{let}\, r = op(v_1, ..., v_n)\, \textbf{in}\, s \xrightarrow{\tau} \{v/r\}s}\ \text{OP}$$

$$\boxed{p \xrightarrow{l} p'}$$

$$\frac{s \xrightarrow{l} s'}{\textit{tid}{:}s \xrightarrow{\textit{tid}:l} \textit{tid}{:}s'}\ \text{THREAD} \qquad \frac{p_1 \xrightarrow{l} p_1'}{p_1|p_2 \xrightarrow{l} p_1'|p_2}\ \text{PAR1} \qquad \frac{p_2 \xrightarrow{l} p_2'}{p_1|p_2 \xrightarrow{l} p_1|p_2'}\ \text{PAR2}$$

$$\boxed{p, m \xrightarrow{l} p', m'}$$

$$\frac{p \xrightarrow{\textit{tid}:\mathbf{W}x=v} p'}{p, m \xrightarrow{\textit{tid}:\mathbf{W}x=v} p', m \oplus \{x \mapsto v\}}\ \text{SWR} \qquad \frac{m(x) = v \quad p \xrightarrow{\textit{tid}:\mathbf{R}x=v} p'}{p, m \xrightarrow{\textit{tid}:\mathbf{R}x=v} p', m}\ \text{SRD} \qquad \frac{p \xrightarrow{\textit{tid}:\tau} p'}{p, m \xrightarrow{\textit{tid}:\tau} p', m}\ \text{STAU}$$

Say p, m has a *data race* if there is a sequence of transitions $p, m \xrightarrow{l_1} \ldots \xrightarrow{l_n}\xrightarrow{l}\xrightarrow{l'}$ where l and l' *conflict*: they are reads or writes to the same location, at least one is a write, and they are by different threads.

[continued . . .]

(*a*) Give a p for which p, m_0 has a data race. [1 mark]

(*b*) A *vector clock* c is a function from thread ids to natural numbers, identifying the $c(tid)$'th transition of each thread tid. Modify the semantics above to add a vector clock c to each process thread ($tid_c{:}s$), each process label ($tid_c{:}l$), and each memory location (with each $m(x)$ now being a pair v_c of a value and vector clock). In your semantics each vector clock should be computed so as to record the latest transition number of all threads that have causally affected that point. Explain your semantics, perhaps with some simple examples. [11 marks]

(*c*) Suppose that $p, m \xrightarrow{l}\xrightarrow{l_1} \ldots \xrightarrow{l_n}\xrightarrow{l'}$ in your vector-clock semantics, where l and l' conflict but are separated by $l_1, \ldots, l_n$. To implement a dynamic race detector, we would like to find conditions on $l_1, \ldots, l_n$ under which there is some other execution with l and l' adjacent: $p, m \xrightarrow{\hat{l}_1} \ldots \xrightarrow{\hat{l}_{\hat{n}}}\xrightarrow{\bar{l}}\xrightarrow{\bar{l}'}$ (where $\bar{l}$ and $\bar{l}'$ are like l and l' but perhaps with different vector clocks). Give such a condition, as liberal as you can, and explain why it has that property. [8 marks]

10 Semantics of Programming Languages

Consider the language L below, with call-by-value functions, ML-style references, and types $\mathbf{nat}_+$ and $\mathbf{real}_+$ of positive natural and positive real numbers. L includes a primitive test for primality, $\mathbf{prime}\,(e)$, and a square-root function, $\mathbf{sqrt}\,(e)$; these are defined only for positive-natural and positive-real values respectively.

$$T ::= \quad \mathbf{bool} \mid \mathbf{nat}_+ \mid \mathbf{real}_+ \mid T \to T' \mid T\,\mathbf{ref}$$
$$e ::= \quad x \mid n \mid r \mid \mathbf{fn}\,x : T \Rightarrow e \mid e\ e' \mid \mathbf{ref}\ e \mid\ !e \mid e := e' \mid \mathbf{prime}\,(e) \mid \mathbf{sqrt}\,(e)$$

Here x ranges over a set X of variables and n and r range over $\mathbb{N}_{>0}$ and $\mathbb{R}_{>0}$ respectively. Let Γ range over finite partial functions from X to types T.

(*a*) Give typing rules defining $\Gamma \vdash e : T$ for $\mathbf{prime}\,(e)$ and $\mathbf{sqrt}\,(e)$. [1 mark]

(*b*) There is an obvious runtime coercion from elements of $\mathbf{nat}_+$ to elements of $\mathbf{real}_+$. To let programmers exploit that conveniently, we would like to define a type system for L that includes a subtype relation $T_1 <: T_2$ with $\mathbf{nat}_+ <: \mathbf{real}_+$. The type system should prevent all run-time errors.

(*i*) Give the other rules defining $T_1 <: T_2$ and the subsumption rule to use that relation in $\Gamma \vdash e : T$. [4 marks]

(*ii*) Give the 6 (standard) typing rules defining $\Gamma \vdash e : T$ for functions and references. [3 marks]

(*iii*) With reference to your subtype rule for function types, explain covariance and contravariance of subtyping. Give examples in L showing that your rule is the only reasonable choice. [2 marks]

(*iv*) Similarly, justify your rule for reference types. [2 marks]

(*c*) To implement L, we want to translate it during typechecking to another typed language L′ which makes that coercion explicit where required, as a new expression form $\mathbf{real\_of\_nat}(e)$, and which does not have subtyping.

(*i*) Give the L′ typing rule for $\mathbf{real\_of\_nat}(e)$ and indicate any other changes required to your type rules for L. [1 mark]

(*ii*) Define an inductive relation $T <: T' \rightsquigarrow e$ which for any $T <: T'$ constructs a coercion $e : T \to T'$. [4 marks]

(*iii*) Define an inductive relation $\Gamma \vdash e \rightsquigarrow e' : T$ where e is an L expression and e' is an L′ expression which is like e but with coercions introduced where needed, such that $\Gamma \vdash e : T$ iff $\exists e'.\ \Gamma \vdash e \rightsquigarrow e' : T$. You should explain but need not prove that, and you can omit the rules for references. [3 marks]

END OF PAPER