COMPUTER SCIENCE TRIPOS Part IB

Tuesday 5 June 2007 1.30 to 4.30

PAPER 4

*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
Script paper
Blue cover sheets
Tags

SPECIAL REQUIREMENTS
None

1 Concurrent Systems and Applications

(*a*) What is meant by a *serializable* order for two or more transactions? [2 marks]

(*b*) Explain how *timestamp ordering* (TSO) enforces isolation. [5 marks]

(*c*) Draw and explain a history graph for two transactions whose invocations of a set of conflicting operations are serializable but which are rejected by TSO. [5 marks]

(*d*) Considering Optimistic Concurrency Control (OCC):

(*i*) State the properties of a transaction's set of shadow copies that must be verified at commit time. [2 marks]

(*ii*) Carefully explain the algorithm used by a single-threaded commit-time validator. [4 marks]

(*e*) Consider a system in which the transactions cause updates to objects which are not all located on a single server but which are distributed widely around the Internet. What factors would influence your choice of using TSO or OCC to enforce isolation? [2 marks]

2 Probability

Suppose you have k light bulbs, where $k > 1$, and that the probability of any individual bulb not working is p. Two strategies for testing the k bulbs are:

(A) Test each bulb separately. This takes k tests.

(B) Wire up all k bulbs as a series circuit. If all the bulbs come on, the testing is complete in just one test, otherwise revert to strategy A taking a total of $k+1$ tests.

Let X be a random variable whose value r is the number of tests required using strategy B. The probability $\mathrm{P}(X = r)$ may be expressed as:

$$\mathrm{P}(X = r) = \begin{cases} (1-p)^k, & \text{if } r = 1 \\ 1-(1-p)^k, & \text{if } r = k+1 \\ 0, & \text{otherwise} \end{cases}$$

(a) Explain this function and justify the constraint $k > 1$. [4 marks]

(b) Determine the Expectation $\mathrm{E}(X)$. [4 marks]

(c) Strategy B beats strategy A (by requiring fewer tests) if $\mathrm{E}(X) < k$ and this condition is satisfied if $p < f(k)$ where $f(k)$ is some function of k. Derive the function $f(k)$. [8 marks]

[Note that $f(k) \to 0$ as $k \to \infty$ and that the maximum value of $f(k) \approx 0.307$ (when $k = 3$). Strategy B is therefore never useful if $p > 0.307$.]

(d) Suppose you have n light bulbs, where $n \gg k$ and k divides n so that $n = m.k$, and you partition the n bulbs into m groups of k. Assuming that the groups are independent and again assuming that $k > 1$, show that the expected number of tests is:

$$n\big[1 + \frac{1}{k} - (1-p)^k\big].$$

Give a rough description of how, for a given value of p, the expression in square brackets varies with k and suggest how someone responsible for testing light bulbs might exploit this expression. [4 marks]

3 Prolog

Short Message Service (SMS) texts replace lists of three characters with single characters to be able to represent the information in as few characters as possible when typing on a phone keyboard. For example, "See you later, Kate" becomes `C u l8r k8`. This question asks you to create predicates in Prolog to implement this translation and to describe how they work using the examples given.

Your answers should use minimal backtracking but should achieve this without using the cut operator.

(*a*) Write a predicate `replace(OldCharacter, NewCharacter, InputList, OutputList)` that replaces all the occurrences of `OldCharacter` in `InputList` with `NewCharacter`. For example, `replace(a,x,[b,a,n,a,n,a], Answer)` unifies `Answer` with `[b,x,n,x,n,x]`. [2 marks]

(*b*) Explain how your `replace` program produces this output, showing carefully how and when backtracking and unification occur. [2 marks]

(*c*) Describe the *two* circumstances where the cut operator is recommended when using Prolog as a "pure" logic language. [2 marks]

(*d*) Write a predicate `textify(ListToReplace, NewCharacter, InputList, OutputList)` that replaces all the occurrences of `ListToReplace` in `InputList` with the character `NewCharacter`. Assume that `ListToReplace` always has exactly three characters.

For example, `textify([a, t, e], 8,[s, e, e, ' ', y ,o ,u , ' ', l, a, t, e, r, ' ', k, a, t, e], Answer)` should unify `Answer` with `[s, e, e, ' ', y, o, u, ' ', l, 8, r, ' ', k, 8]`. [5 marks]

(*e*) Explain how your `textify` program produces this output, showing carefully how and when backtracking and unification occur. [4 marks]

(*f*) Provide the calls to `textify` to replace `[a,t,e]` with `8`, `[s,e,e]` with `c` and `[y,o,u]` with `u` for the `InputList [s, e, e, ' ', y ,o ,u , ' ', l, a, t, e, r, ' ', k, a, t, e]`. [3 marks]

(*g*) Describe how you would modify `textify` to deal with lists of any length. [2 marks]

4 Compiler Construction

(a) Lexical analysis is an important first step in compilation.

(i) Define in detail the abstract "lexing problem" that is solved by a lexical analyser. [4 marks]

(ii) Describe in detail how a lexical analyser can be automatically constructed from a list of regular expressions. [2 marks]

(b) Explain why LL(1) parsing is associated with left-most derivations of parse trees. [6 marks]

(c) Context-free grammars and ambiguity.

(i) Define when a context-free grammar is *ambiguous*. [2 marks]

(ii) A *left-recursive* context-free grammar production has the form $A \to A\alpha$. In the same way, a *right-recursive* production has the form $B \to \beta B$. (Both α and β are assumed to be non-empty sequences.) Suppose we have a grammar that contains both a left- and a right-recursive production for the same non-terminal, such as $A \to A\alpha$ and $A \to \beta A$. Is such a grammar always ambiguous? [6 marks]

5 Mathematical Methods for Computer Science

(a) Suppose that X_n is an irreducible Markov Chain with transition matrix P and suppose that π is a probability distribution over the states of the Markov Chain.

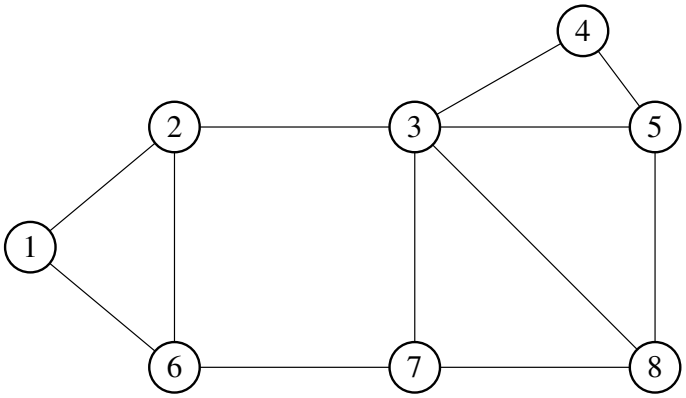
(i) State the detailed balance conditions for the Markov Chain to be reversible in terms of the distribution π. [4 marks]

(ii) Show that if the detailed balance conditions hold then π is a stationary distribution for the Markov Chain. [4 marks]

Suppose that G is a graph with vertices $i \in N$ and undirected edges $(i, j) \in E$ where N and E are both finite sets. Assume that G is connected so that there is a path between every pair of nodes. Define a Markov Chain on the graph G with states given by the vertices, N, and transition matrix P such that $P_{ij} = 1/v_i$ if $(i, j) \in E$ and $P_{ij} = 0$ otherwise where v_i is the number of edges incident at vertex i ($i \in N$).

(b) Show that $\pi_i = v_i / \sum_{j \in N} v_j$ for $i \in N$ is in detailed balance with P. [6 marks]

(c) Suppose that G has eight vertices and undirected edges as shown in the figure:



Find the stationary distribution for the Markov Chain on G and determine the relative proportions of time spent at each of the eight vertices. [6 marks]

6 Computation Theory

(a) What does it mean for a set of natural numbers $S \subseteq \mathbb{N}$ to be

(i) *recursive*? [1 mark]

(ii) *recursively enumerable*? [2 marks]

(b) Show that if a set is recursive, then it is also recursively enumerable. [5 marks]

(c) Let ϕ_e denote the partial function from $\mathbb{N}$ to $\mathbb{N}$ computed by the register machine with code $e \in \mathbb{N}$. Is either of the following sets of numbers recursively enumerable? Justify your answer in each case, stating clearly any standard results that you use.

(i) $S_1 = \{e \in \mathbb{N} \mid \text{for all } x \in \mathbb{N},\ \phi_e(x) \text{ is defined}\}$. [6 marks]

(ii) $S_2 = \{e \in \mathbb{N} \mid \text{for some } x \in \mathbb{N},\ \phi_e(x) \text{ is defined}\}$. [6 marks]

7 Artificial Intelligence I

A very simple neural network designed to solve a two-class classification problem where the classes are labelled as 0 and 1 takes input vectors $\mathbf{x}^T = (\ x_1\ \ x_2\ \ \ldots\ \ x_n\ )$ and has a weight vector $\mathbf{w}^T = (\ w_1\ \ w_2\ \ \ldots\ \ w_n\ )$, both with real-valued elements. It computes the function

$$f(\mathbf{w};\mathbf{x}) = \text{sgn}\left(\mathbf{w}^T\mathbf{x}\right) = \text{sgn}\left(\sum_{i=1}^{n} w_i x_i\right)$$

where the function sgn is defined as

$$\text{sgn}(z) = \frac{1}{1+e^{-z}}.$$

There exists a training sequence for the network containing m labelled examples

$$((\mathbf{x}_1, o_1), (\mathbf{x}_2, o_2), \ldots, (\mathbf{x}_m, o_m))$$

where the o_i denote desired outputs and take values in $\{0, 1\}$.

(a) For the given training sequence, the error of the network when the weights are set to $\mathbf{w}$ is to be defined by the function

$$E(\mathbf{w}) = \lambda||\mathbf{w}|| + \sum_{i=1}^{m}\left(o_i \log\frac{1}{f(\mathbf{w};\mathbf{x}_i)} + (1-o_i)\log\frac{1}{1-f(\mathbf{w};\mathbf{x}_i)}\right)$$

where λ is a fixed, real-valued parameter, we use natural logarithms, and $||\mathbf{w}|| = \sum_{i=1}^{n} w_i^2$. Derive an algorithm that can be used to train this neural network by attempting to find a weight vector minimizing $E(\mathbf{w})$. [17 marks]

(b) Describe the way in which your algorithm might be affected by applying it using different values for the parameter λ, in particular very large or very small values. [3 marks]

8 Introduction to Security

(*a*) Your colleague wants to use a secure one-way hash function h in order to store h(password) as password-verification information in a user database for which confidentiality might become compromised. For h, she suggests using an existing CBC-MAC routine based on AES with all bits of the initial vector and the 128-bit AES key set to zero. Is this construct a suitable one-way hash function for this application? Explain why. [8 marks]

(*b*) Explain how, and under which circumstances, overlong UTF-8 sequences could be used to bypass restrictions regarding which files an HTTP server serves. [8 marks]

(*c*) Name *four* techniques that can be used to make buffer-overflow attacks more difficult. [4 marks]

9 Algorithms II

(*a*) Define *four* of the following terms, stating their defining properties and making use of equations where appropriate.

(*i*) flow network (a graph $G = (V, E)$);

(*ii*) flow (a function $f : V \times V \to \mathbb{R}$);

(*iii*) value of a flow (a real number);

(*iv*) residual network (a graph $G_f = (V, E_f)$);

(*v*) residual capacity (a function $c_f : V \times V \to \mathbb{R}$);

(*vi*) augmenting path (a sequence of edges).

[4 marks]

(*b*) Give some clear pseudocode for the Ford–Fulkerson method of finding the maximum flow and discuss its running time. Prove that, under appropriate conditions (which ones?), the method terminates. [4 marks]

(*c*) Given a flow network $G = (V, E)$ and two flows f_1 and f_2 in G, let $f_3 : V \times V \to \mathbb{R}$ be defined as

$$f_3(x, y) = f_1(x, y) + f_2(x, y).$$

Is f_3 a flow in G or not? Give a full proof of your answer, with reference to the three properties of a flow. [4 marks]

(*d*) Explain what a maximum matching in a bipartite graph is and explain how to solve it by transforming it into a maximum flow problem. [2 marks]

(*e*) Let $G(V, E)$ be a bipartite graph, with the vertex set V partitioned into a left subset L and a right subset R, and edges going from L to R. Let G' be the corresponding flow network according to the construction you explained in part (*d*). Derive a reasonably tight upper bound for the number of edges in any augmenting path that may be discovered by Ford–Fulkerson on G'. [6 marks]

END OF PAPER