PRIFYSGOL CYMRU; UNIVERSITY OF WALES

DEGREE EXAMINATIONS MAY/JUNE 2003

SWANSEA

Computer Science

CS 132 Algorithms and Computation

Attempt 2 questions out of 3

Time allowed: 2 hours

Students are permitted to use the dictionaries provided by the University

Students are NOT permitted to use calculators

CS 132 ALGORITHMS AND COMPUTATION

May/June 2003

(Attempt 2 questions out of 3)

Question 1

(a) Consider the following array

$$A = A[1 \ldots 7] = \mathrm{S, O, R, T, I, N, G}$$

with respect to the usual alphabetical ordering. Represent the main stages in sorting this array with

(i) InsertionSort(A). (Sequence of insertions.)

(ii) MergeSort(A). (Splitting pattern and sequence of merges.)

(iii) HeapSort(A). (Main steps in Heap(A) and in the main procedure.)

Give a brief description in your own words, of the essential features of each sorting strategy.

[12 marks]

(b) Consider sorting in a setting where a large sorted array needs to be maintained in sorted order in response to individual updates of single entries (one at a time say). Which of the above three sorting procedures — InsertionSort, MergeSort, or HeapSort — offers the best choice in this setting? Justify your choice with a careful statement as to the appropriate complexity guarantee for the chosen algorithm in this setting.
[N.B.: You are *not* asked to modify or adapt the chosen basic sorting algorithm for use in the given scenario.]

[5 marks]

(c) (i) State the worst-case complexity for the above three sorting procedures — InsertionSort, MergeSort and HeapSort — with respect to the number of comparisons needed to sort any array of length n. Also rank these growth rates.

(ii) Someone claims to have found a comparison based sorting algorithm that needs no more than $n + 1$ comparisons to sort any array of length n, for every $n \geqslant 1$.

Expose this impostor: state the lower bound result that shows this claim to be false, and adapt the decision tree argument to this concrete setting; in particular, find some small concrete array length n for which the impostor's claim must break down, and argue the case for this n.

[8 marks]

Question 2

(a) Let $\Sigma = \{a, b, c\}$ and consider the following deterministic finite automaton (DFA) $\mathcal{A}$:

$Q = \{0, 1, 2, 3\}$
$q_0 = 0$
$A = \{2, 3\}$

δ	a	b	c
0	1	0	0
1	1	2	1
2	1	3	2
3	1	0	3

(i) Draw the transition diagram of $\mathcal{A}$.

(ii) Which of the following Σ-words are accepted by $\mathcal{A}$?
Justify your answers.

$$W_1 = cabacabb, \quad W_2 = caabacab, \quad W_3 = cabbca.$$

(iii) Describe in words the language accepted by $\mathcal{A}$.

[9 marks]

(b) (i) Argue that no DFA with fewer than 4 states can accept the same language as the above DFA $\mathcal{A}$. To this end show that the runs on the following 4 words *necessarily* have to terminate in 4 distinct states, for *any* DFA that recognises the desired language:

$$\epsilon, \quad a, \quad ab, \quad abb.$$

(ii) Give an example of some simple language that is not recognised by any finite automaton, and explain how this can be shown.

[6 marks]

(c) (i) Discuss how Turing machines crucially differ from finite automata.
[Hint: you may think of which specialisations it would take to restrict the Turing machine model to make it no more powerful than finite automata, from DTM to DFA say.]

(ii) Describe the Halting Problem and discuss its significance with respect to the limitations of the algorithmic method in principle.

[10 marks]

Question 3

Consider deterministic finite automata (DFA) and non-deterministic finite automata (NFA) over the alphabet $\Sigma = \{a, b\}$.

(a) Consider the DFA $\mathcal{A} = (\Sigma, Q, q_0, \delta, A)$ $\quad Q = \{0, 1, 2, 3\}$, $q_0 = 0$, $A = \{0\}$

δ	a	b
0	1	0
1	2	3
2	3	0
3	3	3

(i) Draw the transition diagram of $\mathcal{A}$.

(ii) Trace the runs of $\mathcal{A}$ on inputs $W_1 = aababa$ and $W_2 = aabbbaabb$ and determine whether these words are accepted.

(iii) Describe in words and by means of a regular expression the language $L(\mathcal{A})$ that is accepted by $\mathcal{A}$.

[9 marks]

(b) Consider the NFA $\mathcal{B} = (\Sigma, Q, q_0, \Delta, A)$ depicted in the diagram below

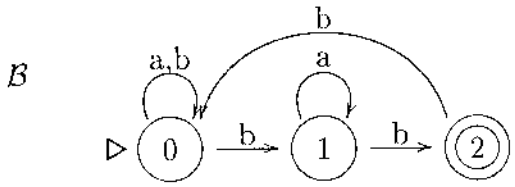
(i) Point out why this transition diagram is *not* that of a DFA.

(ii) Trace all the runs of $\mathcal{B}$ on input $W = ababbb$ and determine whether W is accepted.

(iii) Apply the power set construction to obtain a DFA $\mathcal{B}^{\text{det}} = (\Sigma, \hat{Q}, \hat{q}_0, \delta, \hat{A})$ that accepts precisely the same language as $\mathcal{B}$.

(iv) Describe in words or by means of a regular expression the language $L(\mathcal{B})$ that is accepted by $\mathcal{B}$.

[10 marks]



(c) (i) Construct automata $\tilde{\mathcal{A}}$ and $\tilde{\mathcal{B}}$ that accept precisely the complements of the languages accepted by $\mathcal{A}$ and $\mathcal{B}$, respectively.
($\mathcal{A}$ and $\mathcal{B}$ the DFA and NFA from parts (a) and (b), respectively.)

(ii) Why is it that the construction of an automaton for the complement language is not as straightforward for NFA as it is for DFA? Illustrate in a concrete example the observation that the simple trick that works for DFA does not in general apply to NFA. [E.g., consider the above $\mathcal{B}$.]

[6 marks]