ENGINEERING TRIPOS PART IIA

Monday 2 May 2005 2.30 to 4

Module 3I1

DATA STRUCTURES AND ALGORITHMS

*Answer **all** of Section A and **two** questions from Section B.*

All questions carry the same number of marks.

*The **approximate** percentage of marks allocated to each part of a question is indicated in the right margin.*

There are no attachments.

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

(TURN OVER

SECTION A

Attempt all parts of this question

1. You should answer each of the following parts with a short answer, quoting relevant results rather than proving them. Give names for algorithms you need to refer to and sketch methods or justifications of claims you make. Credit will be given for the clarity and succinctness of answers as well as for basic factual accuracy.

(a) What is a splay-tree and how do you look up data in one? [10%]

(b) Name a data-structure suitable for implementing a priority queue, and give an indication of how much time it should take to remove the top item from it. [10%]

(c) Give an example of a useful algorithm whose computing time cost can be characterised as $O(n^{20})$. [10%]

(d) Explain briefly what is meant by "amortised" computing cost estimates and comment why they may sometimes be more realistic than simple worst-case analysis. [10%]

(e) Give a set of operations and identities that you would expect to form an Abstract Data Type for a priority queue. [10%]

(f) Suggest circumstances (for instance an application, programming language being used, performance demands or other constraints) where you would use free-store allocation based on a Buddy system rather than using Garbage Collection. [10%]

(g) Indicate circumstances where Garbage Collection would probably be better than use of the Buddy system. [10%]

(h) Perfection in a binary tree would see all leaves at the same height. Red-Black trees do not achieve perfection, even though they do avoid extremes of failure to balance the tree. How out of balance can a Red-Black tree get? [10%]

(cont.

(i) Describe an algorithm that can find the minimum spanning tree of an arbitrary graph or show that no such tree exists. You do not have to prove its correctness of estimate its costs. [10%]

(j) Knuth reported that the sequence generated by $a_i = a_{i-24} + a_{i-55} \pmod{2^{32}}$ can give a reasonable sequence of pseudo-random values provided that the first 55 values in the sequence (which are needed to start things off) are not all even. In what way might the sequence fail to behave randomly if all the first 55 values are even? [10%]

(TURN OVER

SECTION B

2. How would you sort data in each of the following circumstances? Justify or comment on your choices of method.

(a) The data is a set of 10 entries in the high-table list for a video-game console, to be sorted by score. [14%]

(b) You have 10 million people to sort based on their age in years. If two people have the same age it does not matter which order they appear in the output list. [14%]

(c) You have 10 million values but rather than wanting them totally sorted you just want a list of 101 numbers (a_0 to a_{100}) where the value a_k in your output would have been k% of the way through the full list if that had been sorted. [14%]

(d) You have just 1000 items to sort, but it takes 20 seconds or so to compare any pair of them. [14%]

(e) Every second you receive a large file of numbers over the network, and you must display the top 100 (in this batch) in descending order. If very occasionally you can not complete preparing this list of the sorted top 100 on time it does not matter. [14%]

(f) As (e) but you need the top 1000 and you will lose your job if the required data from one set of numbers can not be displayed before the next set of data arrives. [15%]

(g) Your data consists of a large file of names. The first (about) 90% of it is already believed to be in sorted order from last time (but you are not 100% confident about that). The remaining 10% is new data that has been written onto the end of the original file in a chaotic unordered state. You want a single file consisting of the old and new data all neatly sorted. [15%]

3. Consider the message "amanaplanacanalpanamaΩ", where the "Ω" signifies the end of the message. Observe for instance that the letter "a" occurs 10 times.

(a) Based on the frequencies of letters used in the message, show how to construct a Huffman code that will compress it well. In your code how many bits will be used to encode the letters "a" and "Ω"? [25%]

(b) What information does a decoder need in order to expand a message based on your Huffman code? You do not need to discuss how it will obtain this information. [25%]

(c) Huffman coding maps each character of its input onto as few bits as it can. How many bits will it take to pack the first four characters of the sample message here, and what will be the exact sequence of bits that it uses? [25%]

(d) How many bits would Lempel-Zif compression need to send the first four characters of this message, supposing that it knew in advance that it was having to deal with messages based on an alphabet of 6 characters plus an end of file marker? Explain why Huffman (that reduces the number of bits per symbol sent) and L-Z (that increases it) can both eventually lead to good compression. [25%]

4. A DVD can store around 4 Gbytes of information. You are charged with designing a way of storing a large dictionary on a DVD. There will be around 500,000 words included and each word has an associated description that may be several thousand characters long. You may suppose that data can be read from the DVD at around 5 Mbytes per second, but a "seek" to a new location on the DVD takes around a fifth of a second.

(a) Explain what benefit Larsen's Dynamic Hashing could have for organising the data on your disc. Describe how you would organise the DVD into blocks, how you would create the data in the form to be written to it, and how the program to read the DVD would locate desired dictionary entries. Estimate the time it would take to look up a given word on your dictionary-disc. [35%]

(b) As an alternative to Larsen's method, discuss B-trees. Explain and estimate the same things you did in section (a). [35%]

(c) Although in the short term your dictionary will always be accessed directly off its DVD, you imagine that in the near future your users will have enough disc space that they copy all the data from your DVD onto their hard disc to get faster access to it. At that stage you will offer a service of network-delivered corrections and updates. In the light of this, and considering any effects that might arise as your dictionary grows so that it almost totally fills the DVD, discuss the relative merits of the two approaches. [30%]

END OF PAPER