ENGINEERING TRIPOS PART IIA

Wednesday 6 May 2009 9:00 – 10:30

Module 3I1

DATA STRUCTURES AND ALGORITHMS

*Answer **all** of Section A (which consists of short questions), 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.*

STATIONERY REQUIREMENTS	SPECIAL REQUIREMENTS
Single-sided script paper	none

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

SECTION A

Answer all parts of this question. The question in this section will be marked out of the same total as each question in section B, and each part carries the same weight.

1 (a) Larsen's dynamic hashing is sometimes used when data lives on disc, and it manages to guarantee that retrieving data only ever uses a single disc access. Suppose my data is kept in memory not on disc, but I arrange that memory in blocks so that Larsen's method means I only access my target memory block just once. Is using Larsen in this way liable to be competitive with other in-store options? Is the cost of accessing information $O(1)$? [10%]

(b) Imagine a variation of quicksort that selects two pivots, say p and q, at random with $p < q$. It then partitions the data into values less than p, between p and q and a third block for data greater than q. It recurses on each sub-block. Give a recurrence formula for the cost in the ideal case where the partition splits the data into three equal-sized regions. Is the ideal cost still proportional to $n\log(n)$? [10%]

(c) Give a very brief sketch of an algorithm to find a minimum spanning tree for a graph. You do not need to include either a proof of its correctness or an analysis of its costs. [10%]

(d) What statistical expectation about text-files does the Lempel Zif compression method (as used in "zip") rely on to allow it to compress data? How might you construct data that did not satisfy this expectation? [10%]

(e) Explain what is meant by "amortised" cost analysis, giving an example from the realm of data structures and algorithms where it is helpful. [10%]

(f) Explain how you could add a single new item into a heap (that is "heap" as in "heapsort"). [10%]

(g) Explain what aspects of computing costs are *not* captured when you specify costs using big-O notation. Why is this often found useful and how does big-Theta

(cont.

notation provide a variation? [10%]

(h) What problem does Ephemeral Garbage Collection (sometimes known as Generation-based collection) seek to address as a potential problem in other garbage collection strategies? What observation or assumption does it make about the patterns of storage allocation and release in order to achieve a benefit? [10%]

(i) A simple implementation of quicksort is given data where all the values to be sorted happen to be identical. Explain how quicksort behaves in this case, giving enough details of the variant you are describing to back up your explanation, and predict the costs. [10%]

(j) Are the following statements true, and comment on the line of reasoning: (a) Quicksort runs in time $O(n^3)$. (b) MergeSort runs in time $\Theta(n\log(n))$. (c) Big-O and Big-Θ analysis almost always gives predictions that are relevant in the real world. (d) So you should almost always use merge-sort in preference to quicksort. [10%]

SECTION B

*Attempt **two** questions from this section. Each question has the same weight in marks.*

2 Explain the facilities needed in a free-store management system, and comment on the different trade-offs offered by *first fit*, *buddy* and *garbage collection* strategies. [35%]

Two particular garbage collection methods are *mark and sweep* and *stop and copy*. For each of these explain the stages and steps involved, constraints that they impose on the programming system that uses them. Identify the parameters that determine the cost of a single garbage collection. Explain a case where *stop and copy* would complete its collection significantly faster than *mark and sweep*. [30%]

Suppose your application has about N units of data alive at any one stage, and that the computer on which it is run has M units of memory available to be used (it had better be the case that $N < M$). If some aspect of the growth-rate of the cost of one of the garbage collection procedures is $O(n)$ then use a specific symbol, say K_1 for the associated constant of proportionality.

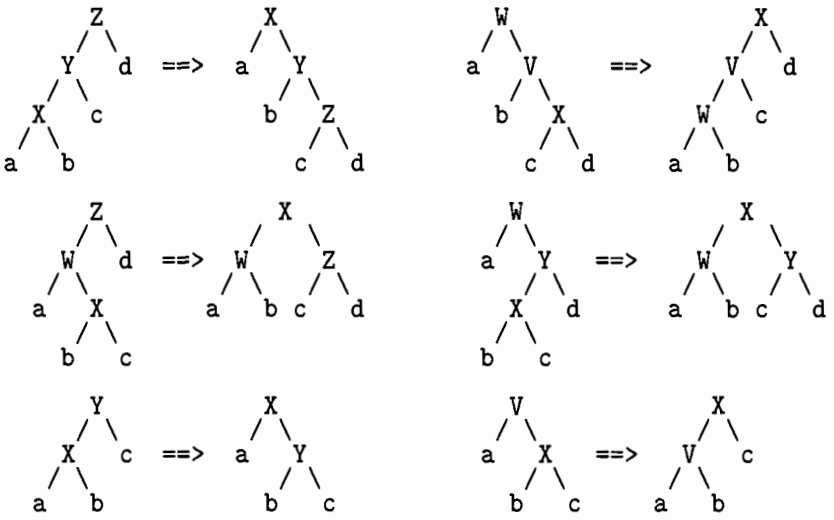
In terms of these constants of proportionality and the ratio between M and N analyse which method is likely to be more economical. [35%]

3 Dijkstra's algorithm for the single-source shortest path problem is often used in the context of routing in computer networks. Give an outline of just what problem it solves, how it works and any constraints on the graph that may need to apply. [30%]

A the graph has V vertices and E edges. The vertex to which you are trying to find path turns out to be as remote from the source as possible. Consider an implementation of the algorithm that identifies each next vertex to process by merely scanning through all the vertices. Do not use any form of heap or priority queue, but you may store and update information in the vertices themselves. In terms of big-O notation how does your cost depend on V and E. A graph is called "dense" if it has (almost) as many edges as any graph with that many vertices can have. How many edges can a graph with V vertices have? Hence express your cost prediction merely in terms of V. [30%]

Now suppose that you have a "sparse" graph (one with many fewer edges than a dense graph would have). Re-analyse your algorithm now supposing that you keep all vertices in a normal Heap, and use the heap as a priority queue to track which vertex to process next. You may assume without further discussion that of a heap contains N items at some moment that the item with smallest value is instantly available, and that new items can be added, old ones removed or the numeric weight associated with an item in the heap updated in time proportional to $\log(N)$. For dense graphs is your new method an improvement on the previous one? A Fibonacci Heap is a data structure whose details you are not required to understand, but which can be used to create a priority queue where removing the top item from the queue has logarithmic cost and all other operations have unit cost (in the amortised sense). Would its use change matters? [40%]

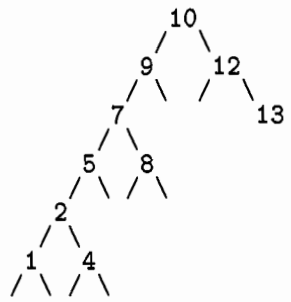
4 The lecture notes contained the following set of diagrams relating to operations on Splay Trees:



Explain when and how these transformations are used and how the choice of which of them to use is made. [25%]

Explain the key properties that Splay Trees then exhibit and comment when they may be useful and when some other structure (which you should also describe at least briefly) would be preferable. [25%]

Starting with the following seriously unbalanced tree, show what happens when you use Splay Tree procedure and access the node containing the value 1. [25%]



Show what the result would have been if you only used the simpler rotations from the bottom row of the original table of transformations. [25%]

END OF PAPER