Math 176 - Fall 2000 - Advanced Data Structures

Midterm Study Topics

Instructor: S. Buss

Here are some topics that may be covered on the midterm, and a few sample questions.

1. Asymptotic analysis. Big-O, little-o, θ, Ω. This topic will definitely be on the midterm, as it is important!

Sample questions: Let $f(n)$, $g(n)$ be functions from the following list: the constant function 1, the constant function 5, $\log n$, $\log^2 n$, $\sqrt{\log n}$, n, $\sqrt{n}$, n^2, n^3, $n \log n$, $n \log^2 n$, 2^n, $n \log n / \log\log n$, etc. Is $f = O(g)$? Is $f = o(g)$? Is $f = \theta(g)$? Is $f = \Omega(g)$?

More sample questions: Suppose $f = O(g)$ and $g = o(h)$. Must $f = o(h)$ hold? Must $f = O(h)$ hold? Must $f = \theta(h)$ hold? Must $f = \Omega(h)$ hold? Must $h = \Omega(f)$ hold? For each question, if the answer is "No", give examples of f, g, h which provide a counterexample.

2. We have studied a variety of data structures. These include arrays (sorted and unsorted), linked lists (sorted and unsorted, singly linked, doubly linked, circularly linked), stacks, queues, deques, binary search trees, AVL trees, threaded AVL trees, B-trees, splay trees, hashing with separate chaining, hashing with linear probing, hashing with quadratic probing, double hashing, and binary heaps.

For each of these data structures, you should know what methods they support efficiently. You should be prepared to give a description of the ADT usually supported by each of these. You should be able to give runtime bounds for each operation on each data structure.

Sample question: For each of the data structures above, how long does a `find(Object o)` operation take? (Use big-O notation). Similarly, how long does a `remove(Object o)` assuming the position of the object `o` has already been found.

3. For each data structure above and for each method that might reasonably be implemented in the data structure, you should be prepared to give pseudo-code describing how to implement the method.

Sample question: Give pseudo-code implementing `deleteMin()` in heaps. Be sure to return the right value. Helper routines (like `percolateUp` or `percolateDown` that are invoked by your pseudo-code must be implemented as well.

Sample question: Write a recursive routine (in pseudo-code) which determines the size (the number of nodes) of a binary search tree.

4. For what purposes are the various data structures useful? When would you choose one data structure over another?

Sample question: which of the above data structures can be used to sort n items in $O(n \log n)$ time? Give pseudo-code for the sorting methods (with your pseudocode just calling the methods of the ADT's, not implementing them).

Sample question: what are the relative advantages and disadvantages of hashing with linear probing versus hashing with quadratic probing?

5. Describe the way data is stored internally for each of the data structures above.

Sample question: For a (non-threaded) Avl tree, describe the data fields stored each node.

6. Be able to carry out simple operations by hand.

Sample question: given a tree in some configuration: what tree would result after a right rotation was performed at node x. Or, what tree would result after key k is inserted into the tree? Or, the same question for an AVL tree.

What would the result of a `buildHeap` operation on a given list of elements?

7. There will be no proofs on the midterm, but you should be able to give numeric bounds or estimates based on theorems proved in class, in the text or in the homeworks.

Sample questions: What is the maximum/minimum number of nodes in a binary tree of height n? What is the maximum number of leaves in a tree with n nodes?

If n objects are inserted into a hash table with separate chaining, and if m is the size of hash table (so n/m is less than the load factor), what is the worst-case maximum number of elements in the list pointed to the hash table entry? What is the average number of elements in the lists?