

## Study guide for the final exam

Exam date: May 6.

This is a small organizational guide to the material in the course and the final exam. First I list the three main categories, then I refine them. Of course, this list may not be complete and the categories are not exclusive. Good luck

- I. Mathematics and orders of magnitude.
- II. Basic algorithms and data structures: what they are and what they are good for
- III. Be able to combine algorithms and data structures to solve more complex problems efficiently.

### Specifics for these, a partial list.

- I.
  - Order of magnitude terminology and notation:  $\Theta$ ,  $\Omega$ , big “O”, little o.
  - geometric, arithmetic, harmonic ( $\sum 1/k$ ) sums.
  - estimation of sums by integrals and simple bounds. Estimating sums where the terms themselves are estimated using  $\Theta$ , etc.
  - formulating and solving recurrence relations – unrolling, finding  $b_k = a_{2^k}$  if  $a_n$  satisfies a recurrence relation. Formulating a recurrence relation from a recurrent algorithm, recurrence relations for quicksort and mergesort.
- II. Data structures: arrays, pointers, linked lists, queues, stacks, trees (binary search trees, B trees (just what they are and what they can do, not the complex insertion and deletion algorithms), red black trees (same), search trees for a graph), graphs (directed, undirected, DAG, incidence list, incidence matrix, edge list), heaps, hash tables, ...  
Algorithms: sorting, heaping, hashing, tree building and balancing, union/find (and amortized analysis stuff), graph traversals and finding topological structure of a graph – uses of BFS and DFS. Greedy algorithms (including Huffman). Dynamic programming and applications. MST and shortest path algorithms.
- III. Review the various ways we combined data structures in more more complex algorithms. This is often done using threading pointers. Examples:
  - A structure where you can delete the item with minimum key or the item with a given name (combine hash table and heap with pointers).

- A two ended heap (deleteMax, deleteMin) using threaded heaps, or deleteMin by one key or another (thread two min heaps).
- Search a small part of a large graph using a hash table to store which vertices you have visited.