CS 513
Midterm 1

Prof. Farach-Colton Oct 7, 2019

You may consult your 1 page crib sheet. Any theorem used in class or algorithm presented in class or homeowrk can be used. Other theorems or algorithms can only be used if you prove them. 25% credit for clearly stating “I don’t know.” -25% for copying out a proof or algorithm that we’ve already done in class (that is, don’t waste your time copying an old proof or algorithm just to have filler. If we already have an algorithm, you can use it as a black box.). For all algorithmic questions, the faster the algorithm, the more points (assuming that you’ve got a correct algorithm!). And you need to analyze any algorithm you describe.

1. **Word algorithms.** Give the run time of each, assuming that all integers are have $\log n$ bits, and that that is the size of the word. (So XOR, +, time, AND, ... are all constant time.) If you decide to do some preprocessing, seperately analyze the running time of the preprocessing and the query.

   (a) Give an algorithm for picking out the ith bit of a number. That is, $\text{PICKBIT}(x, i)$ returns 1 if the ith bit of x is 1, and 0 otherwise.

   (b) Give an algorithm for finding the least significant 1 in a word. That is, LS1(x) returns i, if the $\text{PICKBIT}(x, i) = 1$ and, for all $j < i$, $\text{PICKBIT}(x, j) = 0$.

   (c) Given two words, x and y, give an algorithm to compute $\vee_{i=1}^{\log n} x_i \wedge y_i$, where $x_i = \text{PICKBIT}(x, i)$, and similarly for y_i.

2. **van Emde Boas meets hashing.** Although van Emde Boas trees support all dictionary operations efficiently, they are extremely big. In a RAM with word size w, the space cost is $\Theta(2^w) = \Theta(u)$. This is because every $\text{vEB}(u)$ node uses an entire array of size $\sqrt{u}$ to store children pointers, even if only a few subtrees are nonempty, where $\text{vEB}(u)$ is a vEB-tree that can store elements in the set $\{0, \ldots, u - 1\}$.

   A *space-efficient van Emde Boas tree* replaces each array by a cuckoo hash table. You will recall that a cuckoo hash table performs insertions in expect $O(1)$ time and lookups in worst case $O(1)$ time, and uses $O(n)$ space to store n elements.

   (a) What is the running time of each operation in a space-efficient van Emde Boas tree?

(b) Let $S(u)$ be the space used by a single arbitrary element in a space-efficient $\text{vEB}(u)$ tree. Argue that $S(u) \leq 2\,S(\sqrt{u}) + O(1)$. What is the total expected space cost of the tree, as a function of the total number of elements n, the word size w, and S?

(c) Show that $S(u) = O(\log u)$.

(d) Explain how to bring the space cost down to $O(n)$ without compromising the running time of any operation.

3. **Sorting.** Choose the best sorting sorting algorithm for each case.

   (a) Sort 50TB worth of data stored in the cloud, from your laptop. Your cloud storage plan provides minimal computational power.

   (b) Sort an array with n records, each composed of several data fields, and the order is given by an awful legacy comparison function you refuse to understand.

   (c) Sort n integers in a supercomputer with large word size.

   (d) Sort the n vertices of a graph, by degree.

4. **Strings.** Given a string $S \in [n]^n$, give an algorithm to find the longest palindrome in the string.