Design and Analysis of Algorithms September 18, 2015
Massachusetts Institute of Technology 6.046/18.410
Shafi Goldwasser, Dana Moshkovitz, and Nir Shavit Problem Set 2

Problem Set 2

This problem set is due at **11:59 pm** on **Wednesday, September 23th, 2015**. The exercises are **optional**, and should not be submitted.

Exercise 2- 1: Exercise 9.3-1 in CLRS

Exercise 2- 2: Exercise 9.3-7 in CLRS

Exercise 2- 3: Exercise 20.3-1 in CLRS

Exercise 2- 4: Exercise 20.3-4 in CLRS

Exercise 2- 5: Exercise 20.3-6 in CLRS

Problem 1- 1: Finding the k-th smallest element

Given two sorted arrays of size n with distinct elements, $A[1 \ldots n]$, $B[1 \ldots n]$, find the k^{th} smallest element in the union of the two arrays. You can index into the array and compare elements of the array in $O(1)$ time. Assume that $k <= n$.

(a) Describe and analyze the running time of an algorithm to find the k^{th} smallest element in $O(k)$ time.

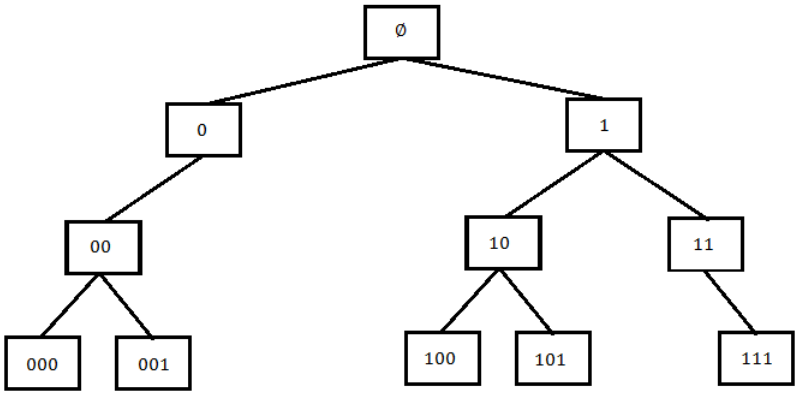
(b) Describe a way to identify $\frac{k}{2}$ of the k smallest elements in $O(1)$ time. Note: It doesn't have to be the smallest $\frac{k}{2}$ elements, just any $\frac{k}{2}$ out of the smallest k elements.

(c) Using the results from part (b), describe and analyze the running time of an algorithm to find the k^{th} smallest element in $O(\log k)$ time.

Problem 2- 1: Structures related to van Emde Boas

In this problem you will explore another data structure, closely related to van Emde Boas, that achieves the $O(\log \log U)$ running time for all operations. In this problem we will concentrate on the SUCCESSOR operation.

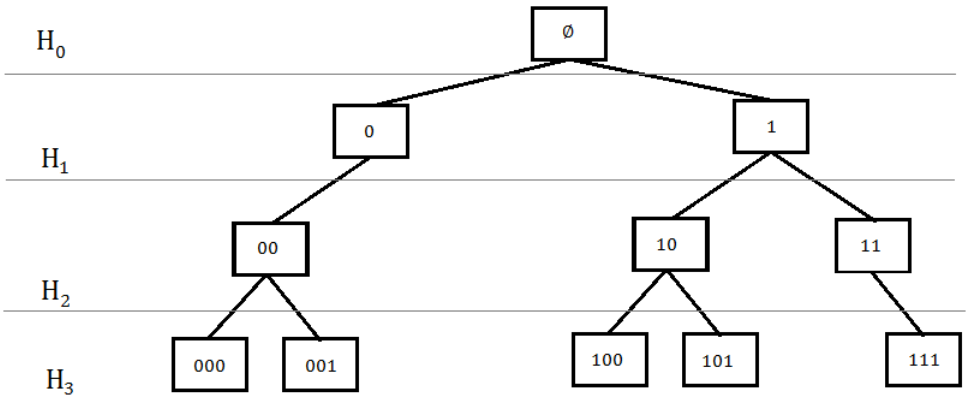
A trie with binary number keys is a binary search tree, where INSERT, FIND and REMOVE all take $O(\log U)$ time. U is the universe size, in our case $U = 2^m$, so the structure can store integers in range $0..2^m - 1$. All keys are placed only at the bottom level of the structure (in the leaves). A parent represents a binary prefix of its (one or two) children. An example for $U = 8$, with keys $0, 1, 4, 5$ and 7 inserted:



You can read more about the trie structure on https://en.wikipedia.org/wiki/Trie

(a) A successor of the key x is the smallest integer y in the structure, such that $y > x$. Describe an algorithm to find the SUCCESSOR of key x in $O(\log U)$ time.

(b) Let's augment each level of the trie with a structure that provides INSERT, FIND and REMOVE in $O(1)$ time (for example, a hash structure), for all binary strings at that level. You can assume that these functions return a pointer to the corresponding node in the trie. Since binary strings at k^{th} level have length k, each level has at most 2^k different strings. For example, you could query $H_2(01)$ or $H_3(101)$ in $O(1)$.



What are the running times of INSERT, FIND and REMOVE now?

(c) Assume that you are given the structure from b) augmented with the $O(1)$ hash structures at each level. Additionally, each node stores the minimum and the maximum value in the sub-trie it represents.

Describe an algorithm to find SUCCESSOR in $O(\log \log U)$ time.