CS240E S26	Tutorial 8	July 3

Q1. Maximum XOR.

Suppose you have an array $A[1..n]$ of n elements, each an integer within the range $0 \leq A[i] < 2^l$. Give an algorithm to find the maximum of $A[i]$ xor $A[j]$ for $1 \leq i, j \leq n$ in $O(nl)$.

Q2. Consecutive strings.

Given an uncompressed trie T that stores a list of binary strings, design an algorithm $\mathit{consecutive}(b_1, b_2)$ that takes two binary strings in T as input, and outputs true if the strings are consecutive in pre-order traversal of the trie, and outputs false otherwise.

Assume that branches are ordered as $\$, 0, 1$. The runtime should be bounded by $O(|b_1| + |b_2|)$.

For example, suppose T stores $\{000, 01, 0110, 101, 11\}$. Then:

- $\mathit{consecutive}(0110, 101)$ returns true
- $\mathit{consecutive}(01, 000)$ returns true
- $\mathit{consecutive}(11, 000)$ returns false

Q3. MSD-radix sort as a trie.

Consider the following base-4 numbers:

$$300, 211, 112, 230, 1, 0, 12, 101, 233, 110.$$

(a) Draw the recursion tree that results from sorting these numbers with MSD-radix. Note that it is a 4-way pruned trie.

(b) Show that the expected time to insert a base-4 number into a 4-way pruned trie is at most $\log_4 n + O(1)$, assuming all numbers have been uniformly chosen. You may assume the numbers have been padded with 0s so that all numbers begin with the same place value.

(c) Suppose we have n English words (26-letter alphabet), where the combined length of all words is l. Give an algorithm to sort the words (obtain lexicographical ordering) in $O(l)$ time.

Q4. Ordered Hashing.

For this question we will look at a modified version of open addressing such that, while searching in the hash table, the sequence of keys encountered during the search has some ordering property (though the sequence is not necessarily in sorted order). To

achieve this property, we use a modified insert routine: when inserting key k, follow its probe sequence, and if you ever encounter a key k' with $k' > k$, then swap to put k into this position, and proceed to insert k'.

(a) Carry out this idea for linear probing using $M = 10$, the hash function $h(k, i) = (k + i) \mod 10$, and the insertion sequence: $31, 26, 16, 23, 11, 30, 20$.

(b) Argue that for searching for any key, the keys encountered before it in its probe sequence are all smaller than it. You may assume that no items will ever be deleted from the hash table.

(c) Give an improved search method for ordered hashing.

Q5. Uniform hash functions. Recall, a family $\mathcal{H}$ of hash functions is *universal* if

$$P(h(k) = h(k')) \leq \frac{1}{M} \quad \text{for all keys } k \neq k'.$$

Recall further that $\mathcal{H}$ has *uniform hash-values* if

$$P(h(k) = i) = \frac{1}{M},$$

for all keys k and all slots i. The probability is taken over the random uniform choice of h among $\mathcal{H}$. Note that uniform hash values does **not** imply a universal hash function.

1. Consider the family $\mathcal{H}$ on slide 4 of module07e:

$$U = \mathbb{Z}_5, M = 2$$

$$h_b(k) = ((k + b) \mod 5) \mod 2$$

$$\mathcal{H} = \{h_b : b \in \mathbb{Z}_5\}$$

   Choose $b \in \mathbb{Z}_5$ randomly to get hash-function. Does this have uniform hash-values? Is this universal?

2. Assume that $\mathcal{H}$ has uniform hash-values. Prove or disprove: The expected time for an unsuccessful search in hashing with chaining is $O(\alpha)$.

3. Assume that $\mathcal{H}$ has uniform hash-values. Prove or disprove: The expected time for a successful search in hashing with chaining is $O(1 + \alpha)$.