CS 240: Data Structures and Data Management **Fall 2020**

Final Help Session

Reminder: Final Exam is between December 15, 4 pm and December 16, 4 pm.

Note: This is a sample of problems designed to help prepare for the final exam. These problems do *not* encompass the entire coverage of the exam, and should not be used as a reference for its content.

1 True/False

For each statement below, write true or false.

a) Open addressing hashing that uses linear probing will require two hash functions.

b) Run-length encoding may result in text expansion on some strings.

c) When doing range search on a quadtree, if there is no point within the range specified, the worst case runtime complexity is in $\Theta(h)$.

d) Suffix trees for pattern matching require preprocessing the pattern.

e) Inserting a set of keys into an empty compressed trie will always result in the same final trie regardless of the insertion order.

f) The runtime complexity of range query for kd-trees depends on the spread factor of points.

g) When using KMP to search for the pattern $\texttt{a}^{m-1}\texttt{b}$ in the text $\texttt{a}^{m}$, the positions of the pattern shifts are the same as the brute-force algorithm.

h) Rehashing may be required in Cuckoo Hashing even if the load factor is at an acceptable value.

i) Every AVL Tree is also a 2-4 Tree.

j) Move-to-front transformation uses adaptive instead of static dictionaries.

2 Multiple Choice

Pick the best answer for each question.

a) Which of the following functions $f(i)$ would cause interpolation search to have the least worst-case runtime on an array A with $A[i] = f(i)$?

   a) $f(i) = \log(i)$
   b) $f(i) = i$
   c) $f(i) = i^2$
   d) $f(i) = 2^i$

b) Given $h_0(k) = k \bmod 7$ in a hash table of size 7, which of the following hash functions would be most suitable for h_1 in double hashing?

i) $h_2(k) = k^2 \bmod 7$

ii) $h_2(k) = (k \bmod 6) + 1$

iii) $h_2(k) = 2 \cdot (k \bmod 4)$

iv) $h_2(k) = \left\lfloor \frac{1}{2} \cdot (k \bmod 13) \right\rfloor$

c) Given $h_0(k) = k \bmod 7$ with two hash tables, each of size 7, which of the following hash functions would be most suitable for h_1 in cuckoo hashing?

i) $h_2(k) = k^2 \bmod 7$

ii) $h_2(k) = (k \bmod 6) + 1$

iii) $h_2(k) = 2 \cdot (k \bmod 4)$

iv) $h_2(k) = \left\lfloor \frac{1}{2} \cdot (k \bmod 13) \right\rfloor$

d) If the root of a quadtree represents the region $[0, 128) \times [0, 128)$ while the deepest (lowest) internal node represents the region $[88, 92) \times [24, 28)$, what is the height of the quadtree?

a) 4

b) 5

c) 6

d) 7

e) Which one of the following statements about compressed tries is false?

i) Every internal node stores an index indicating the bit position to be tested on a search.

ii) The root of the compressed trie always tests the first bit.

iii) A compressed trie that stores n keys always contains less than n internal nodes.

iv) The height of a compressed trie never exceeds the length of the longest string it stores.

f) Which of the following search operations on a non-dictionary structure has the most efficient worst-case runtime?

i) Searching for a specific key in a max-heap.

ii) Searching for a specific point in a kd-tree with points in general position.

iii) Searching for any occurrence of a specific character in a text using a suffix tree, with children pointers stored as arrays.

iv) Searching for a specific character in a decoding trie of characters (like Huffman's Trie)

g) CS240 is a course about

i) Data structures and algorithms

ii) Unreasonable time management

iii) Reconsidering academic/life choices

iv) All of the above

3 Pattern Matching

Consider the problem of searching for the pattern $P =$ `OMNOMNOM` in the text $T =$ `OMNOONOMNEMOMNOMNOM` with the alphabet $\Sigma = \{O, M, E, N\}$.

a) Construct the failure array for P. Then use KMP to search for P in T, showing each character comparison. Indicate characters that are known to match due to the failure array using brackets.

b) Construct the last-occurrence function and suffix-skip array for P. Then use Boyer-Moore to search for P in T, showing each character comparison. Indicate characters that are known to match using brackets: use round brackets for matches due to the suffix-skip array, and square brackets for matches due to the last-occurrence function.

You may use the following table as a template. Use a separate table for each sub-problem, and add rows whenever necessary.

O	M	N	O	O	N	O	M	N	E	M	O	M	N	O	M	N	O	M

4 Huffman Compression

a) The following message was compressed using Huffman encoding and transmitted together with its dictionary:

00100001110101011100010110100 10

Char	(space)	: (colon)	d	ℓ	p	s	u	w
Code	100	1011	1010	010	001	000	11	011

Decompress the string using the dictionary and write the final message.

b) Agent Kang doesn't know the information in the message beforehand, but upon seeing the decoded string, he immediately realizes that the message has been tampered with. Explain how Frank determined this.

5 Encoding/Decoding `1`s

Consider the string $S =$ `11111111` (eight `1`s).

a) Use Run-Length Encoding to encode S.

b) Use Run-Length Decoding to decode S.

c) Use Lempel-Ziv Welch to encode S, using the ASCII Table as the initial dictionary. Note that the ASCII value of `1` is 49.

d) Use Burrows-Wheeler Transform to encode S, treating the end-character `$` as coming before `1` in lexicographical order.

6 KD-Trees

Consider the following set of points (ordered by x-coordinates):

$$(2, 90), (22, 70), (42, 84), (55, 96), (58, 26),$$
$$(63, 80), (70, 15), (74, 3), (81, 50), (90, 32).$$

a) Draw the kd-tree corresponding to these points.

b) Indicate on this tree exactly which nodes are visited during a range query in the rectangular box $[50, 90] \times [90, 100]$.

7 Range Trees

Consider the primary tree (x-BST) of a range tree in Figure 1.

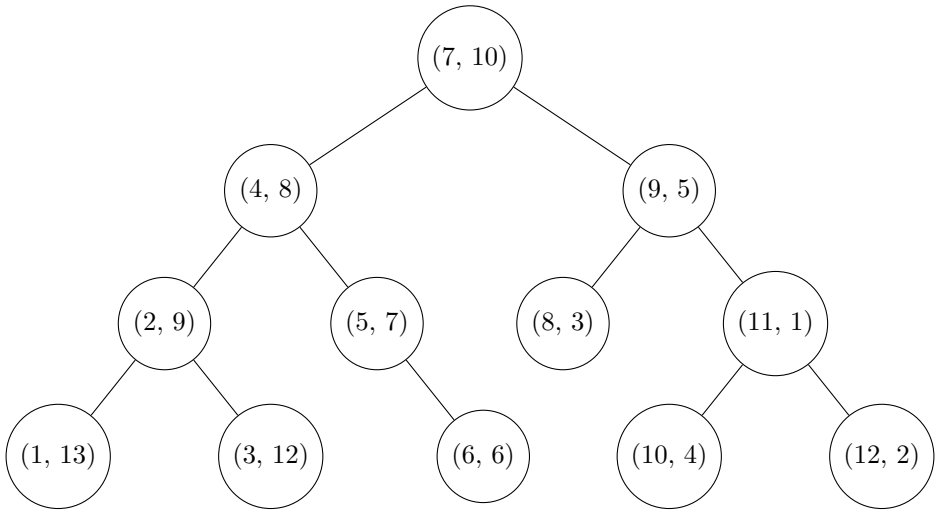

Figure 1: Range Tree x-BST

a) Draw the associated trees (y-BSTs) at nodes $(2, 9)$, $(5, 7)$, and $(9, 5)$.

b) For the query range $R = [0, 7.5] \times [9, 14]$, identify the boundary nodes, inside nodes, and outside nodes for just the x-dimension.

8 Run-Length Encoding

a) Give a string of n bits that achieves the best compression ratio with Run-Length Encoding from all n-bit strings, and state the exact compression ratio achieved.

b) Same question, but for the worst compression ratio. You may assume that n is divisible by 4.

9 Consecutive Trie Strings

Given an uncompressed trie T that stores a list of binary strings, design an algorithm $Consecutive(b_1, b_2)$ 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 $\{000, 01, 0110, 101, 11\}$.

$Consecutive(0110, 101)$ outputs true.
$Consecutive(01, 000)$ outputs true.
$Consecutive(11, 000)$ outputs false.

10 Suffix Trees

Jason discovered a secret message in the form of a Suffix Tree S, indicating the location of a hidden treasure.

a) Design an algorithm that recovers the original text T from its corresponding suffix tree S. The algorithm should run in $O(n)$ time while using $O(n)$ auxiliary space.

b) Determine the original text for the following suffix tree:

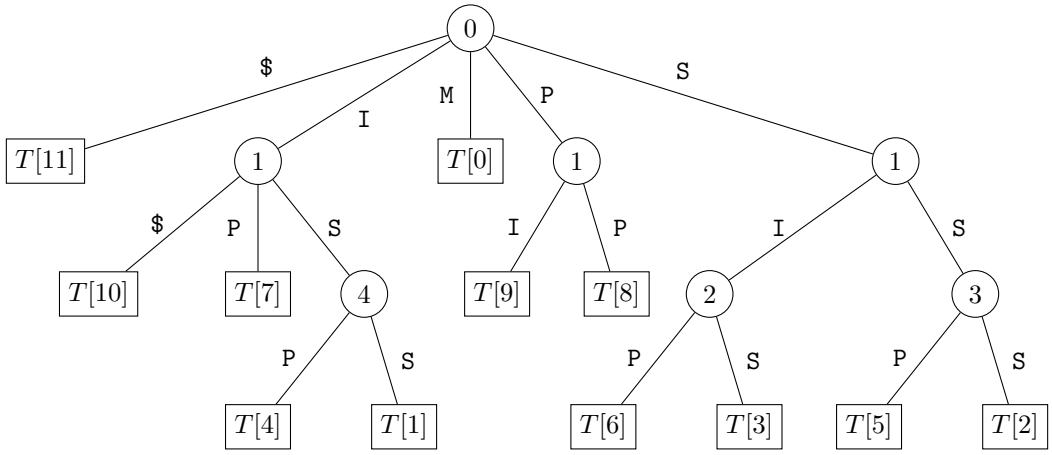

Figure 2: Mysterious Suffix Tree

11 B-Trees

Consider the following B-Tree, of order 5:

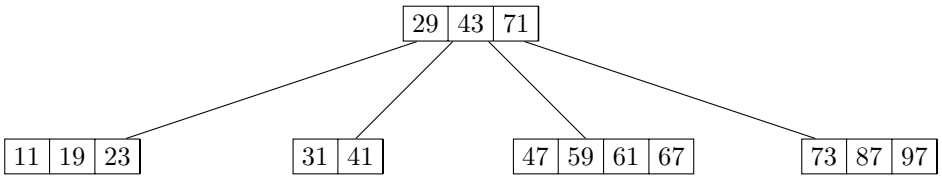

Figure 3: B-Tree of order 5

a) Insert the following keys into the B-Tree, in the order given: 13, 53, 17. Show the tree after each insertion.

b) Delete the following keys from the original B-Tree, in the order given: 19, 43, 31, 29. When deciding between successor/predecessor, choose the successor. When deciding between left or right sibling for transfer/merge, select the right sibling. Show the tree after each deletion.

12 Maximal Difference

Consider an array A of n integers. We want to implement a range query called $MaxDiff(i, j)$ which will find the maximal difference between two elements from $A[i]$ to $A[j]$ inclusive, for $i < j$. For example, suppose our array A is:

$A = 3\ 0\ 5\ 4\ 5\ 6\ 3\ 4\ 5\ 7\ 9\ 8\ 1\ 0\ 1$

If we run the query $MaxDiff(2, 9)$, then the subarray from indices 2 to 9 is:

$A[2 \ldots 9] = 5\ 4\ 5\ 6\ 3\ 4\ 5\ 7$

The largest number is 7 and the smallest number is 3, so the maximal difference is $7 - 3 = 4$. The query $MaxDiff(2, 9)$ should return 4.

Design a data structure for A with space complexity $O(n)$ to answer queries of the form $MaxDiff(i, j)$ in $O(\log n)$ time. There are no constraints on the runtime for preprocessing the array into the data structure.