Comp 3170 - Analysis of Algorithms & Data Structures

Midterm

Shahin Kamalli
University of Manitoba

March 4, 2019

Write your name and student id here:

‘It is not the Mountain Ahead That Wears You Out; It Is the Grain of Sand in Your Shoe’   unknown

. Do not open this booklet until instructed.

• You are allowed use at most one page of printed/written material. Please turn off your cell phones and put them in your bags.

• Manage your time. We start the exam at 10:30 and end the exam at 11:25. You have 55 minutes. Don’t waste too much time on a single question. It is a long exam and your time is limited.

• There are 6 pages (including this cover page). Write your answers in the provided boxes.

• In case that you find the exam too long/hard (which is likely), do not panic. The marks will be scaled so that the highest mark gets the full mark.
1. Short Answer (27 marks)

Provide your short answers in the provided boxes. There is no need to justify your answers. **Notes:** all parts have 2 marks except the last one which has 3 marks.

1. True or False: \( \log_3(n^4) \in \Theta(\log n) \).
   Answer: True
   We have \( \log_3 n^4 = \log n^4 - \log 3 = 4 \log n - \log 3 \in \Theta(\log n) \).

2. True or False: \( \sqrt{n \log n} \in \omega(\log n) \).
   Answer: True
   For any positive values of \( \epsilon \) and \( k \), as long as they are constants (independent of \( n \)), we have \( n^\epsilon \in \omega(\log^k n) \). In particular, \( n^{1/2} \in \omega(\log^2 n) \) which confirms the statement of this question is correct.

3. True or False: It is possible to augment an AVL tree on \( n \) nodes so that rank() and select() operations can be performed in \( O(\log n) \).
   Answer: True
   We saw in the class how to augment AVL trees to support rank and select in \( O(\log n) \).

4. True or False: Height of a balanced binary search tree is \( \Omega(\log n) \).
   Answer: True
   We saw in the class that the height of any binary tree is at least \( \log n - 1 \) (see slide 19 of the binary-search-tree handouts). Intuitively, the height is minimized if the tree is perfectly full and balanced (in this case the height will be \( \log n - 1 \)). Note that the height can be indeed much larger than this.

5. True or False: Quick-select runs in \( O(1) \) in the best case.
   Answer: False
   Regardless of what the input is, we have to read it at least once and it requires \( \Omega(n) \) (e.g., finding the minimum cannot be done in \( O(1) \) in the best case).

6. True or False: Insertion time (time to insert a new item) in Binomial heaps is asymptotically less than the insertion time in ordinary Binary Heaps.
   Answer: False
   The expected height of a skip list of \( n^2 \) items is \( \Theta(n^2) = \Theta(2 \log n) = \Theta(\log n) \). Answer: The advantage of binomial heaps is in the merge operation. Insertion takes \( O(\log n) \) in both binomial and binary heaps.

7. True or False: The expected height of a skip list formed by \( n^2 \) items is \( O(\log n) \).
   Answer: True

8. True or False: if \( f(n) \in \Theta(g(n)) \) then \( 2f(n) \in \Theta(2g(n)) \).
   Answer: False
   Assume \( f(n) = 2n \) and \( g(n) = n \). We have \( f(n) \in \Theta(g(n)) \). But \( 2f(n) = 2^{2n} = 2^n \times 2^n \) which is asymptotically larger than \( c \times 2^n \) for constant \( c \). Here \( 2f(n) \in \omega(2g(n)) \).

9. True or False: Binomial trees are binary trees.
   Answer: False
   Clearly some nodes in binomial trees of order \( \geq 3 \) has more than two children.
10. True or False: It is possible to merge two binomial heaps of size \( n \) in \( O(\log n) \).

**Answer:** True. We can merge binomial heaps in \( O(\log n) \). This was their main advantage over binary heaps.

11. True or false: A binary search tree on \( n \) items can have height \( \omega(n) \).

**Answer:** False. The height is maximized when the tree forms a chain of nodes with only one child. In this case, the height will be \( \Theta(n) \) which is asymptotically less than \( \omega(n) \).

12. True or false: Given \( n \) integers, it is possible to form an AVL tree from them in \( O(n) \).

**Answer:** False. Unlike heaps, binary search tree cannot be formed in \( O(n) \). As we saw in the class, an in-order traversal of a BST gives the sorted sequence of items. Assuming AVL tree can be formed in \( O(n) \), we can sort \( n \) numbers by forming an AVL tree and traversing it. We know, however, sorting requires \( \Omega(n \log n) \).

13. Assume \( T(1) = 5 \) and \( T(n) = 25T(n/5) + n^2 \). Give an expression for the run-time of \( T(n) \) using \( \Theta \) notation.

**Answer:** \( \Theta(n^2 \log n) \). Case two of Master theorem. We have \( n^{\log_5 25} = n^2 \). So we have \( T(n) = n^2 \log n \).
2. AVL Trees (6 marks)

Perform operation \textit{insert}(9) on the following AVL tree. Draw the tree before and after each rotation performed (no need to show balance factors).
3. Skip Lists (8 marks)

1. Show how $\text{Search}(S, 20)$ proceeds in the skip list below. More specifically show what nodes and in which order are visited. You should refer to the nodes using their keys and levels, e.g., you can say “node 104 at level 1” or show them in the picture.

\[ \text{Answer: } (-\infty, 4) \rightarrow (-\infty, 3) \rightarrow (-\infty, 2) \rightarrow (1, 2) \rightarrow (1, 1) \rightarrow (2, 1) \rightarrow (10, 1) \rightarrow (10, 0) \rightarrow (16, 0) \rightarrow (17, 0) \rightarrow (28, 0) \]

Unsuccessful Search.

2. Describe an algorithm that, given an AVL tree with $n$ KVPs, creates a skip list of the same KVPs in expected time $O(n)$. It is sufficient to described the algorithm in a few sentences and provide evidence that it runs in $O(n)$.

\[ \text{Answer: First, use an inorder traversal of the tree to form a sorted list of the KVP's in } O(n). \text{ Given this list, go through all its member from the largest to smallest and insert them to a linked list; this list will form level 0 of the skip list. Since the new item is inserted at the beginning, we do not need to search, and insert takes constant time. Replicate each item (after its insertion) with coin flips; it takes expected constant time per KVP as we saw in the class. So, the total time is expected to be } \Theta(n). \]
4. Binomial Heaps (6 marks)

Consider the following binomial heaps. Show the resulting heap when we apply the operation extract-max(). In case of merging trees, in case there were three binomial trees of the same order, merge the two ‘older’ trees (keep the new tree which is the product of previous merge). Show your work (intermediate steps).
5. More Short Answer (9 marks)

1. Consider the following pseudocode:

```plaintext
foo(n)
1. i ← 0
2. while i < n do
3. k ← i * i * i
4. while k > 1 do
5. k ← k/2
6. i ← i + 1
7. return prod
```

What is the worst-case running time of \textit{foo}(n)?

Express your answer using $\Theta$-notation in terms of $n$, and be as precise as possible.

$\Theta(n \log n)$

**Answer:** Inside the first while loop, we have a total computation of $\Theta(\log k) = \Theta(\log i^3) = \Theta(\log i)$. So, the time complexity will be $\Theta(\log 2) + \Theta(\log 3) + \ldots + \Theta(\log(n - 1)) = \Theta(\log n!)$. In the class, we saw that $\log n! = \Theta(n \log n)$.

2. Give the largest key that can be inserted into the following AVL tree that results in no rotations. If no such key exists write “no key”.

```
49
```

**Answer:** Any value in the range $[41,49]$ will be added to the left of 50 and does not cause a rotation. Larger values will make the tree unbalanced at 50.

3. The following is a \textit{bad} implementation for finding the $n$th Fibonacci number. Using $\Omega$-notation, provide the tightest complexity class (e.g., logarithmic $\Omega(\log n)$, linear $\Omega(n)$, etc.) for the runtime of the algorithm as a function of $n$. No justification is needed.

```
Ω(2^{n/2})
```

**Answer:** We have $T(n) = T(n - 1) + T(n - 2) + c$ for $n > 2$. So, for large values of $n$, it is asymptotically equal to the Fibonacci series, and we have $T(n) > 2T(n/2) > 4T(n/4) > \ldots > 2^{n/2}T(1)$ which implies $T(n) = \Omega(2^{n/2})$. Note that $T(n)$ is asymptotically (and exponentially) smaller than $\Theta(2^n)$. So we cannot write $T(n) \in \Omega(2^n)$. 
\begin{center}
\begin{tabular}{|l|}
\hline
\textit{fibo}(n) \\
1: \textbf{if} \( n \leq 2 \) \textbf{then} \\
2: \hspace{0.5cm} \textbf{return} 1 \\
3: \textbf{else} \\
4: \hspace{0.5cm} \textbf{return} \textit{fibo}(n - 1) + \textit{fibo}(n - 2) \\
5: \textbf{end if} \\
\hline
\end{tabular}
\end{center}