University of Nevada, Las Vegas Computer Science 477/677 Spring 2023

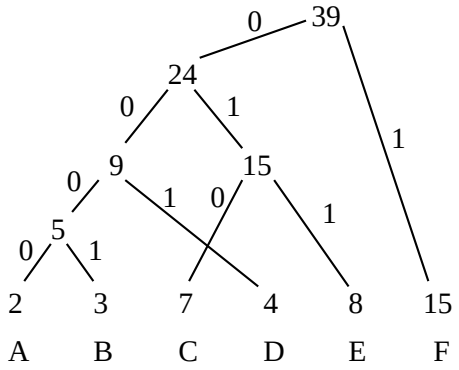
Study Guide for Examination March 8, 2023

1. Review answers to homework3:
   `http://web.cs.unlv.edu/larmore/Courses/CSC477/S23/Assignments/hw3ans.pdf`

2. Review answers to homework4:
   `http://web.cs.unlv.edu/larmore/Courses/CSC477/S23/Assignments/hw4ans.pdf`

3. Use Huffman's algorithm to find an optimal prefix-free binary code for the following weighted alphabet.

A	2	0000
B	3	0001
C	7	010
D	4	001
E	8	011
F	15	1



4. Solve each recurrence using the anti-derivative method.

   (i) $F(n) = F(n-2) + \frac{1}{n}$
   $F(n) = \Theta(\log n)$

   (ii) $F(n) = F(n - \sqrt{n}) + 1$
   $F(n) = \Theta(\sqrt{n})$

   (iii) $F(n) = F(n - \log n) + \log^2 n$
   $F(n) = \Theta(n \log n)$

5. Solve each recurrence using the master theorem.

   (iv) $F(n) = 2F(\sqrt{n}) + \log n$ (Use substitution.)
   $F(n) = \Theta(\log n \log\log n)$

   (v) $F(n) = 3F(n/2) + 1$
   $F(n) = \Theta(n^{\log 3}) = \Theta(3^{\log n})$

   (vi) $F(n) = 4F(n/2) + n^2$
   $F(n) = \Theta(n^2 \log n)$

6. Solve each recurrence using the generalized master theorem.

   (vii) $F(n) = 3F(n/3) + 3F(2n/3) + n$
   $F(n) = \Theta(n^3)$

   (viii) $F(n) = 3F(n/3) + 3F(2n/3) + n^2$
   $F(n) = \Theta(n^3)$

(ix) $F(n) = 3F(n/3) + 3F(2n/3) + n^3$

$F(n) = \Theta(n^3 \log n)$

(x) $F(n) = 3F(n/3) + 3F(2n/3) + n^4$

$F(n) = \Theta(n^4)$

7. Consider the following recursive program for a function F.

```
int F(int n)
 {
  if(n <= 3) return n;
  else return (F(n/2)+F((n+1)/2)F((n+2)/2+F((n+3)/2)+n*n)%8191;
 }
```

(The purpose of `%8191` is to prevent the integers from exceeding the capacity of a standard desktop computer.)

Suppose you wish to find the value of $F(n)$ for some fixed positive integer n. Give asymptotic answers to the following questions.

(a) What is the time complexity of your calculation if you use the recursive code given above?

$T(n) = \Theta(n^2)$

(b) What is the time complexity of your calculation if you use dynamic programming?

$T(n) = \Theta(n)$

(c) What is the time complexity of your calculation if you use memoization?

$T(n) = \Theta(\log n)$

8. Solve the recurrence: $F(n) = F(\log n) + 1$

$F(n) = \Theta(\log^* n)$