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

Answers to Study Guide for Examination October 25, 2023

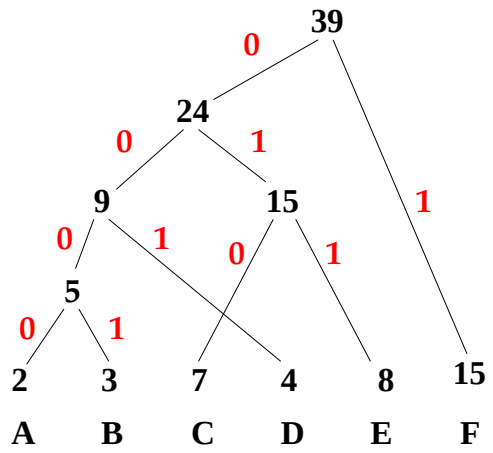
1. Review answers to homework3:
   `http://web.cs.unlv.edu/larmore/Courses/CSC477/F23/Assignments/hw3ans.pdf`

2. Review answers to homework4:
   `http://web.cs.unlv.edu/larmore/Courses/CSC477/F23/Assignments/hw4ans.pdf`

3. Review answers to homework5:
   `http://web.cs.unlv.edu/larmore/Courses/CSC477/F23/Assignments/hw5ans.pdf`

4. 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



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

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

   $F(n) - F(n-5) = \frac{1}{n}$

   $\frac{F(n) - F(n-2)}{2} = \frac{1}{2n}$

   $F'(n) = \Theta(\frac{1}{n})$

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

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

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

   $\frac{F(n) - F(n-\sqrt{n})}{\sqrt{n}} = \frac{1}{\sqrt{n}}$

   $\frac{F(n) - F(n-\sqrt{n})}{\sqrt{n}} = n^{-\frac{1}{2}}$

   $F'(n) = \Theta(n^{-\frac{1}{2}})$

   $F(n) = \Theta(n^{\frac{1}{2}}) = \Theta(\sqrt{n})$

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

$F(n) - F(n - \log n) = \log^2 n$

$\dfrac{F(n) - F(n - \log n)}{\log n} = \log n$

$F'(n) = \Theta(\log n)$

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

6. Solve each recurrence using the master theorem.

`https://en.wikipedia.org/wiki/Master_theorem_(analysis_of_algorithms)`

(iv) Skip this one. I made some mistake: it is not a recurrence problem.
$F(n) = 2\sqrt{n} + \log n$ (Use substitution.)

(v) $F(n) = 3F(n/2) + 1$

$A = 3$, $B = 2$, $C = 0$. $F(n) = \Theta(n^{\log_B A}) = \Theta(n^{\log_2 3})$
since $B^C < A$

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

$A = 4$, $B = 2$, $C = 2$. $F(n) = \Theta(n^{\log_B A} \log n) = \Theta(n^{\log_2 4} \log n) = \Theta(n^2 \log n)$
since $B^C = A$

7. Solve each recurrence using the generalized master theorem, also known as the Akra-Bazzi method.

`https://en.wikipedia.org/wiki/Akra%E2%80%93Bazzi_method`

For all four of the problems below, $k = 2$, $\alpha_1 = \alpha_2 = 3$, $\beta_1 = \frac{1}{3}$, and $\beta_2 = \frac{2}{3}$. We need to find δ such that $\sum_{i=1}^{k} \alpha_i (\beta_i)^\delta = 1$. The solution is $\delta = 2$.

(vii) $F(n) = 3F(n/3) + 3F(2n/3) + n$

In this case, $\gamma = 1$, which is less than δ. Therefore $F(n) = \Theta(n^\delta) = \Theta(n^2)$.

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

In this case $\gamma = 2 = \delta$. Therefore $F(n) = \Theta(n^\delta \log n) = \Theta(n^2 \log n)$.

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

In this case $\gamma = 3 > \delta$. Therefore $F(n) = \Theta(n^\gamma) = \Theta(n^3)$.

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

In this case $\gamma = 4 > \delta$. Therefore $F(n) = \Theta(n^\gamma) = \Theta(n^4)$.

8. 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)
 }
```

We assume that each arithmetic operation takes constant time. The answer to each of the four questions belos is either $\Theta(\log n)$, $\Theta(n)$, $\Theta(n \log n)$, $\Theta(n^2)$, or $\Theta(n^2 \log n)$.

(a) What is the asymptotic complexity of $F(n)$? (Hint: Use the master theorem.)

Asymptotically, $F(n) = 4F(n/2) + n^2$. By the master theorem, $F(n) = \Theta(n^2 \log n)$.

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

(b) What is the time complexity of your calculation if you use the recursive code given above? (Hint: Use the master theorem.)

Let $T(n)$ be the time required to compute $F(n)$. Asymptotically, $T(n) = 4T(n/2) + 1$, hence $T(n) = \Theta(n^2)$.

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

You compute $F(2)$, $F(4)$, $\ldots F(n)$. Each takes $\Theta(1)$ time, so the total time is $\Theta(n)$.

(d) What is the time complexity of your calculation if you use memoization?[1]

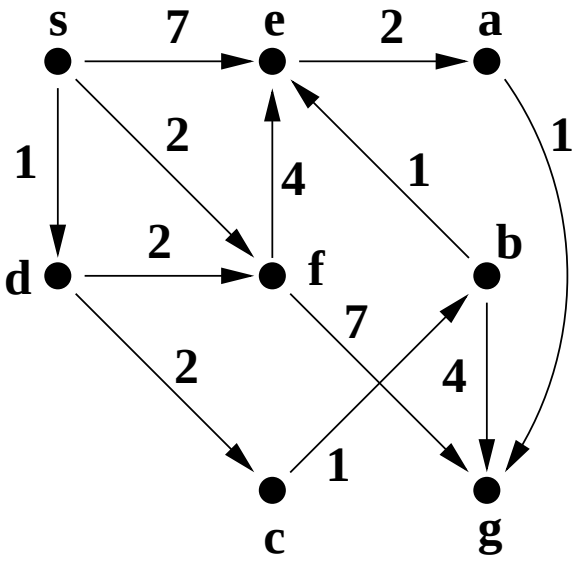
Note: The memos are stored in a search structure. Assume that it takes only constant time to find an item in that structure. That's actually false: the time to find a memo could be the logarithm of the number of memos, but just ignore that factor.
Memoization is explained at "`https://en.wikipedia.org/wiki/Memoization`"
There are $\Theta(\log n)$ memos stored in the structure, hence the answer is $\Theta(\log n)$.

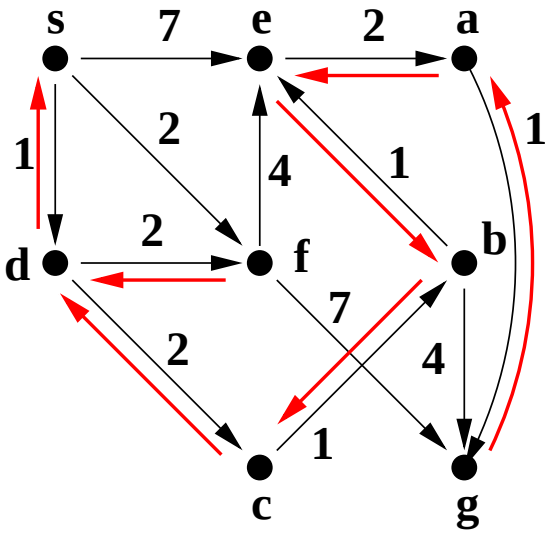
9. Solve the recurrence: $F(n) = F(\log n) + 1$

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

[1] We will cover memoization during the lecture on Monday October 23: if we don't get to it, it won't be on the exam.

10. Walk through Dijkstra's algorithm for the single source minpath problem for the directed graph illustrated below, where **s** is the source vertex.

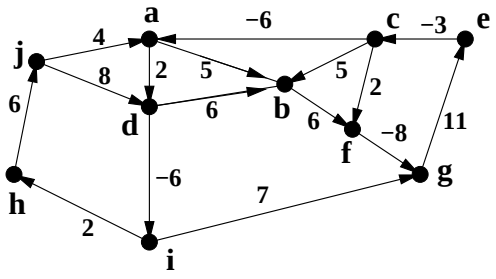




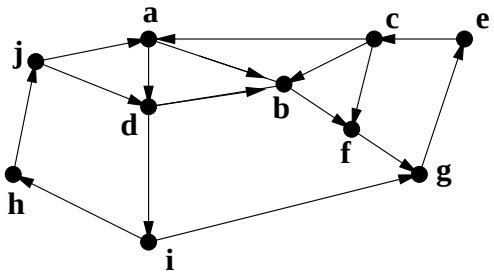
s	a	b	c	d	e	f	g
0	7	4	3	1	~~7~~ 5	2	~~9~~ 8
*	e	c	d	s	~~s~~ b	s	~~f~~ b

11. Figure (a) below shows an instance of the all-pairs minpath problem. Work the first part of Johnson's algorithm on that graph, showing the adjusted weights in Figure (b).

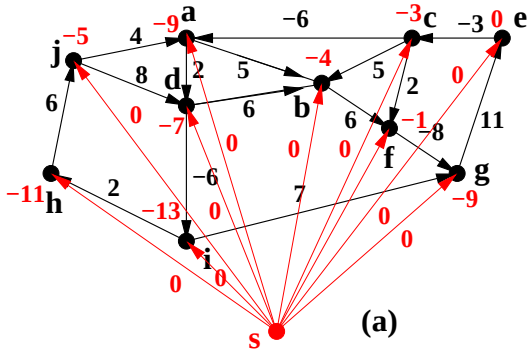
    Do not complete the computation of Johnson's algorithm.



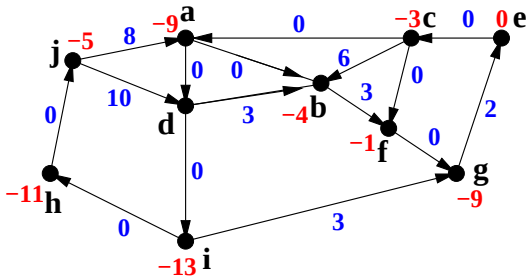
(a)



(b)



(a)



(b)

12. Compute the two square roots of $3 + 4\mathbf{i}$

    Unsing the formula, you get $\pm(2 + \mathbf{i})$.