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

Answers to Examination March 8, 2023

1. Fill in the blanks.

   (a) [15 points] Name three search structures.

   binary search tree
   hash table
   list

   (b) [15 points] Name three priority queues.

   stack
   queue
   heap

   (c) [5 points] Name a divide-and-conquer search algorithm, which only works on a sorted list.

   binary search

   (d) [5 points] Name an $O(n)$-time search algorithm, generally used only when n is small.

   linear search

2. Solve the recurrences, expressing answers using Θ. [10 points each]

   (a) $F(n) = 2F(n/2) + n$
   $A = 2$, $B = 2$, $C = 1$. $F(n) = \Theta(n \log n)$.

   (b) $F(n) = F(\sqrt{n}) + 1$
   SSubstitute $m = \log n$. Then $\log(\sqrt{n}) = m/2$.
   Let $G(m) = F(2^m) = F(n)$
   then $G(m/2) = F(2^{m/2}) = F(\sqrt{n})$
   $G(m) = G(m/2) + 1$
   $F(n) = G(m) = \Theta(\log m) = \Theta(\log \log n)$

3. Find the asymptotic time complexity of each of these code fragments in terms of n, using Θ notation. [10 points each]

   (a)
   ```
   for(int i = 0; i < n; i++)
     for(int j = 1; j < i; j = 2*j);
   ```
   $\Theta(n \log n)$

   (b)
   ```
   for(int i = 1; i < n; i++)
     for(int j = i; j < n; j = 2*j);
   ```
   $\Theta(n)$

   (c)
   ```
   for(float x = n; x > 2.0; x = sqrt(x))
   ```
   (`sqrt(x)` returns the square root of `x`.)

   $\Theta(\log \log n)$

(d) `for(int i = 1; i < n; i = 2*i)`

```
for(int j = 2; j < i; j = j*j);
```

(Hint: use substitution)

$\Theta(\log n \log\log n)$

(e) `for(int i= 0; i < n; i++)`

```
for(int j = 0; j*j < n; j++)
```

$\Theta(n\sqrt{n})$

(f) `for(float i = n; i >= 1.0; i = log(i))`

$\Theta(\log^* n)$

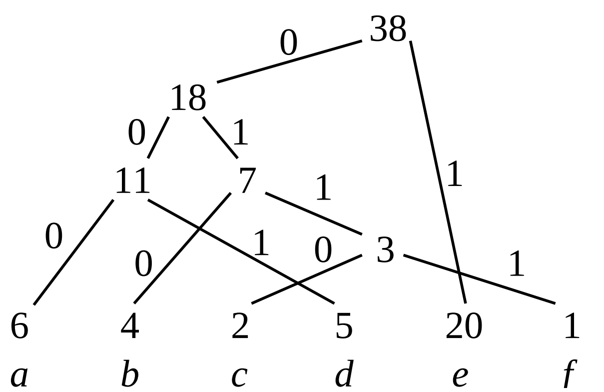
(g) `for(int i = n; i > 1; i = i/2);`

```
for(int j = 1; j < i; j = 2*j);
```

$\Theta(\log^2 n)$

4. [20 points] Find an optimal prefix-free binary code for the following weighted alphabet. Show the Huffman tree.

a	6	000
b	4	010
c	2	0110
d	5	001
e	20	1
f	1	0111



5. [20 points] Compute the Levenstein edit distance from "abcbac" to "cabcba." Show the matrix.

You could use either string to label the rows, and the other to label the columns. Either way is ok, but I'll only give one here.

		a	b	c	b	a	c
	0	1	2	3	4	5	6
c	1	1	2	2	3	4	5
a	2	1	2	3	4	3	4
b	3	2	1	2	3	4	5
c	4	3	2	1	2	3	4
b	5	4	3	2	1	2	3
a	6	5	4	3	2	1	2

The edit distance is 2.

6. Consider the following recursive code, which computes a function $f(n)$.

```
int f(int n)
 {
  if (n <= 0) return 0;
  else return f(n/4) + f(n/4) + f(n/2) + n;
 }
```

(a) [10 points] What is the asymptotic complexity of $f(n)$?

The recurrence is $f(n) = 2f(n/4) + f(n/2) + n$.

Using the generalized master theorem, we obtain $f(n) = \Theta(n \log n)$.

(b) [10 points] What is the asymptotic time to execute the recursive code given above?

Let $T(n)$ be the time to compute $f(n)$. The recurrence is

$T(n) = 2T(n/4) + T(n/2) + 1$

Using the generalized master theorem, we obtain $T(n) = \Theta(n)$.

(c) [10 points] If dynamic programming is used to compute $f(n)$ for a given value of n, what is its asymptotic time complexity?

We compute $f(i)$ for all i from 1 to n. The time complexity is $\Theta(n)$.

(d) [10 points] If memoization is used to compute $f(n)$ for a given value of n, what is its asymptotic time complexity?

The only values of f that need to be computed are $f(n/2^k)$ for $0 \leq k \leq \log n$. It takes $O(1)$ time to compute each one, and thus the time complexity of the memoization algorithm is $O(\log n)$.

I did not take into account the time it takes to execute *insert* and *find* on the seach structure.