[redacted] September 23, 2021

Solutions to Problem 1 of Homework 2 (14+10 points)

[redacted] *Due: 8 am on Thursday, September 23*

[redacted]

Consider the recurrence $T(n) = 2T\left(\frac{n}{2}+1\right) + n - 2$. Further, let $T(3) = 0$. Assume that n is of the appropriate form such that it is always an integer, even as we go down the formula.

(a) (8 points) Prove that $T(n) = \Theta((n-2)\log(n-2))$ using the "guess-then-verify" method. (**Hint**: Recall that if $f(n) = \Theta(g(n))$, then there exists constants $c_1, c_2 > 0$ such that $c_1 g(n) \le f(n) \le c_2 g(n)$ for all $n \ge n_0 > 0$.)

Solution: $T(n) = 2T\left(\frac{n}{2}+1\right) + n - 2$ and $T(3) = 0$
Guess : $T(n) = \Theta((n-2)\log(n-2))$ i.e
$c_1(n-2)\log(n-2) \le T(n) \le c_2(n-2)\log(n-2)$ (from the hint)

Base case $n = 3$:
$T(3) = 0$, in our guess would be,
$c_1(3-2)\log(3-2) \le 0 \le c_2(3-2)\log(3-2)$
$c_1 \log(1) \le 0 \le c_2 \log(1)$
$0 \le 0 \le 0$ which is true in all cases. So we can say that our base case holds for any value of c_1, c_2.

Inductive Step for $n \ge 4$:
Assuming true for $3, 4, 5, ..n-1$ (Inductive Assumption)
Proving for n:
For $T(n) \le c_2(n-2)\log(n-2)$,
$T(n) = 2T(\frac{n}{2}+1) + n - 2 \le 2c_2(\frac{n}{2}+1-2)\log(\frac{n}{2}+1-2) + n - 2$ (inductive assumption)
$\implies 2c_2(\frac{n}{2}+1-2)\log(\frac{n}{2}+1-2) + n - 2 \le c_2(n-2)\log(n-2)$
$2c_2(\frac{n-2}{2})\log(\frac{n-2}{2}) + n - 2 \le c_2(n-2)\log(n-2)$
$c_2(n-2)(\log(n-2) - \log(2)) + n - 2 \le c_2(n-2)\log(n-2)$
$c_2(n-2)\log(n-2) - c_2(n-2) + n - 2 \le c_2(n-2)\log(n-2)$
$-c_2(n-2) + n - 2 \le 0$
$(n-2) \le c_2(n-2)$
$\implies \mathbf{c_2 \ge 1}$ ...(i)

For $c_1(n-2)\log(n-2) \le T(n)$,
$T(n) = 2T(\frac{n}{2}+1) + n - 2 \ge 2c_1(\frac{n}{2}+1-2)\log(\frac{n}{2}+1-2) + n - 2$ (inductive assumption)
$\implies 2c_1(\frac{n}{2}+1-2)\log(\frac{n}{2}+1-2) + n - 2 \ge c_1(n-2)\log(n-2)$
$2c_1(\frac{n-2}{2})\log(\frac{n-2}{2}) + n - 2 \ge c_1(n-2)\log(n-2)$
$c_1(n-2)(\log(n-2) - \log(2)) + n - 2 \ge c_1(n-2)\log(n-2)$
$c_1(n-2)\log(n-2) - c_1(n-2) + n - 2 \ge c_1(n-2)\log(n-2)$

[redacted]

$-c_1(n-2)+n-2 \geq 0$
$(n-2) \geq c_1(n-2)$
$\implies \mathbf{c_1 \leq 1}$ ...(ii)
From (i) and (ii) we know that there exists constants $c_1, c_2 \geq 0$ such that $c_1(n-2)\log(n-2) \leq T(n) \leq c_2(n-2)\log(n-2)$ for all $n \geq n_0(=3) \geq 0$. Hence we can say that our guess holds true and that
$T(n) = \theta((n-2)\log(n-2)$ is true.

□

(b) (**Extra Credit**)(6 points) Solve the same recurrence relation by the domain-range substitution. Namely, make several changes of variables until you get a basic recurrence of the form $R(k) = R(k-1) + f(k)$ for some function f, and then compute the answer from there. This answer will have to be exact and not an asymptotic answer. For full credit, you will have to each of the intermediate substitutions and calculations. (**Hint**: As a first step, you will let $n = c^m + d$ for some constants c, d. What is this choice of c, d?)

Solution: $T(n) = 2T\left(\frac{n}{2}+1\right) + n - 2$
Let $n = 2^k + 2$ (so $c = d = 2$ for $n = c^m + d$) ...(i)
$\implies T(2^k+2) = 2T\left(\frac{2^k+2}{2}+1\right) + 2^k + 2 - 2$
$T(2^k+2) = 2T\left(2^{k-1}+2\right) + 2^k$
Let $S(k) = T(2^k+2)$ ...(ii)
$\implies S(k) = T(2^k+2) = 2T\left(2^{k-1}+2\right) + 2^k$
$S(k) = 2S(k-1) + 2^k$
Dividing throughout by 2^k,
$\frac{S(k)}{2^k} = \frac{2S(k-1)}{2^k} + \frac{2^k}{2^k}$
$\frac{S(k)}{2^k} = \frac{S(k-1)}{2^{k-1}} + 1$
Let $\frac{S(k)}{2^k} = P(k)$ ...(iii)
$\implies P(k) = \frac{S(k)}{2^k} = \frac{S(k-1)}{2^{k-1}} + 1$
$\therefore P(k) = P(k-1) + 1$

Base Case for k = 0, n = $2^0 + 2 = 3$,
So,P(0) = S(0) / $2^0 = T(3) = 0$ (Given)
And, $P(k) = 1 + P(k-1) = 1 + 1 + P(k-2)$
$P(k) = 1 + 1 + 1 + 1 ... 1 + P(0)$
$P(k) = (1)(k) + 0 = k$
On re-substituting values from (ii) and (iii), we get
$P(k) = k = \frac{S(k)}{2^k} = \frac{T(2^k+2)}{2^k}$
$\therefore$T$(2^k+2) = 2^k.k$
Re-substituting value from (i), $n = 2^k + 2$ or $k = \log(n-2)$ we have,
$T(n) = 2^{(\log(n-2))}.\log(n-2)$
$\therefore \mathbf{T(n) = (n-2)\log(n-2)}$
Also, $\implies \mathbf{T(n) = \Theta((n-2)\log(n-2))}$

□

(c) (6 points) Consider the recurrence relation

$$T(n) = 2T(n-1) + n, \; T(0) = 1.$$

Solve by recursion tree method. Towards this end, you will first complete the empty empty fields at this table. You can then conclude that $T(n)$ is the sum of the values in the last column. You will then simplify the sum to give a tight asymptotic bound for $T(n)$, i.e., $T(n) = \Theta(c^n \cdot n^d \cdot (\log n)^r)$ for some constants $c > 1, d, r > 0$.

(**Extra Credit**)(3 points) You get an additional three points if you solve for the value of $T(n)$ exactly and correctly.

(**Hint**: You will find the following formula very useful: Let

$$S(a,d) = \sum_{i=1}^{d} i \cdot a^i = a + 2a^2 + 3a^3 + \ldots + d \cdot a^d = \frac{d \cdot a^{d+2} - a^{d+1} \cdot (d+1) + a}{(a-1)^2} = \Theta(d \cdot a^d)$$

)

Level	Size of Problem	No. of Problems	Non-Recursive Cost of One Problem	Total Cost
0	n	1	n	1.n
1	n - 1	2	n - 1	2.(n-1)
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
i	$n-i$	2^i	$n-i$	$2^i.(n-i)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-1$	1	2^{n-1}	1	$2^{n-1}.(1)$
n	1	2^n	1	$2^n.(1)$

Here, d is the depth of the recursion tree. (**Hint**: To produce a closed form solution)

Solution: For depth d, $n - d = 0$
$d = n$

Total Cost $= 1.n + 2.(n-1) + 2^2.(n-2) + ... + 2^{n-1}.(1) + 2^{(n)}.(1)$
$= 2^0 n + 2^1(n-1) + 2^2(n-2) + 2^3(n-3) + .. + 2^i.(n-i) + ... + 2^{n-1}.(n-(n-1)) + 2^n$
$= \sum_{i=0}^{n-1} 2^i.(n-i) + 2^n$ ..(by observation)

$\therefore$ Total cost $= \sum_{i=0}^{n-1} 2^i.n - \sum_{i=0}^{n-1} 2^i.i + 2^n$
$= n\sum_{i=0}^{n-1} 2^i - \sum_{i=1}^{n-1} 2^i.i + 2^n$ ($\because$ n is a constant in the 1^{st} term and 2^{nd} term is 0 for i=0)
$= n.(1).\frac{2^n-1}{2-1} - \frac{(n-1).2^{n-1+2} - 2^{n-1+1}.(n-1+1)+2}{(2-1)^2} + 2^n$
$= n(2^n - 1) - ((n-1).2^{n+1} - 2^n.(n) + 2) + 2^n$
$= n2^n - n - (2(n-1)2^n - n2^n + 2) + 2^n$

[redacted]

$= n2^n - n - 2n2^n + 2(2^n) + n2^n - 2 + 2^n$

$\therefore$ Total Cost $\mathbf{T(n) = 3(2^n) - n - 2}$

$\implies T(n) = \theta(2^n)$

$\square$

[redacted]

[redacted]ptember 23, 2021

Solutions to Problem 2 of Homework 2 (15+1 Points)

[redacted] *Due: 8 am on Thursday, September 23*

[redacted]

(a) (5 points) Consider the following recurrence:

$$T(n,1) = 3n$$
$$T(1,m) = 3m$$
$$T(n,m) = 3n + T(n/3, m/3)$$

Solve for $T(n,n^2)$ to get a tight asymptotic bound. Assume that n is an exponent of 3 for simplicity.

Extra Credit (1 point): Keep track of the leading coefficient rather than just state $T(n) = \Theta(f(n))$. This means that express $T(n) = c \cdot f(n) + o(f(n))$ for an appropriate choice c, $f(n)$.

Solution: Assumption: $n = 3^a$

$\therefore T(n,n^2) = T(3^a, 3^{2a}) = 3.3^a + T(3^{a-1}, 3^{2a-1})$

$= 3^{a+1} + 3^a + T(3^{a-2}, 3^{2a-2})$

$= 3^{a+1} + 3^a + 3^{a-1} + ... + 3^{a-a+2} + T(3^{a-a}, 3^{2a-a})$

$= 3^{a+1} + 3^a + 3^{a-1} + ... + 3^2 + T(3^0, 3^a)$

$= 3^{a+1} + 3^a + 3^{a-1} + ... + 3^2 + T(1, 3^a)$

$= 3^{a+1} + 3^a + 3^{a-1} + ... + 3^2 + 3.3^a$

$= 3^{a+1} + 3^2 + 3^3 + ... + 3^a + 3^{a+1}$

$= 3^{a+1} + (3^2 + 3^3 + ... + 3^a + 3^{a+1})$ (Sum of GP)

$= 3^{a+1} + 3^2.\frac{3^a-1}{3-1} = 3^{a+1} + \frac{3^2}{2}.(3^a - 1)$

$= 3^{a+1} + \frac{3^2}{2}3^a - \frac{3^2}{2}.1$

$= 3^{a+1} + \frac{3}{2}3^{a+1} - \frac{9}{2}$

$= \frac{5}{2}3^{a+1} - \frac{9}{2} = \frac{15}{2}3^a - \frac{9}{2}$

Re-substituting $n = 3^a$, we get:

$T(n,n^2) = \frac{15}{2}n - \frac{9}{2}$

Here, if f(n) = n, then we can say that $-\frac{9}{2} = o(n)$

$\therefore \mathbf{T(n,n^2) = \frac{15}{2}n + o(n)}$ $\mathbf{(c = \frac{15}{2}, f(n) = n)}$

$\implies \mathbf{T(n,n^2) = \Theta(n)}$ (from Recitation slides)

□

(b) (5 points) Consider the following recurrence

$$T(n) = T(0.01n) + T(0.99n) + cn$$

where $c > 0$ is a constant. Solve for $T(n)$ by the recursion tree method to derive an asymptotically tight solution. (**Hint**: Note that this tree is not going to be a balanced tree. What is the depth of the shortest branch? What is the depth of the largest branch? Use this to derive a Θ bound for $T(n)$.)

Solution:

Level	Size of Problem	Total Cost
0	cn	cn
1	$\frac{cn}{100}, \frac{99cn}{100}$	cn
2	$\frac{cn}{10^4}, \frac{99cn}{10^4}, \frac{99cn}{10^4}, \frac{(99)^2cn}{10^4}$	cn
$\vdots$	$\vdots$	$\vdots\vdots$
i	$\frac{cn}{(100)^i}......(\frac{99}{100})^i cn$	cn
$\vdots$	$\vdots\vdots$	$\vdots$

As we can see from the table, the leftmost branch has the shortest depth while the rightmost branch has the longest depth. We can see the time they take to reach base case ($=1$) would be,

$1 = \frac{cn}{100^{i_{leftmost}}}$

$\therefore 100^{i_{leftmost}} = cn$

$\implies i_{leftmost} = \log_{100} cn$ (i)

Similarly for $i_{rightmost}$,

$1 = (\frac{99}{100})^{i_{rightmost}} cn$

$\implies i_{rightmost} = \log_{100/99} cn$ (ii)

$\implies$ The time it takes to go down the tree is tightly bound to $log(cn)$, (from (i) and (ii)) i.e. $\theta(logn)$

With this, we can say that the total cost of T(n) would be,

$cn. \log_{100} cn \leq T(n) \leq cn. \log_{100/99} cn$

$\implies \Omega(nlogn) \leq T(n) \leq O(nlogn)$

Hence, we can conclude that $\mathbf{T(n) = \theta(nlogn)}$

□

(c) (5 Points) Consider the following recurrence

$$T(n) = 9T(n/3) + \frac{n^2}{\log_3 n}.$$

Solve for $T(n)$ by domain-range substitution. Namely, make several changes of variables until you get a basic recurrence of the form $R(k) = R(k-1) + f(k)$ for some f, and then compute the answer from there. You may assume that n is a power of 3. (**Hint**: Begin by expressing n as a function of another variable m. Then change the variable T. You will change T to a function S and then to R. Note: The two changes can be combined in one step too.)

Solution: Assumption: $n = 3^m$ (n is some power of 3)

$T(3^m) = 9T(\frac{3^m}{3}) + \frac{(3^m)^2}{log_3 3^m}$

$T(3^m) = 9T(3^{m-1}) + \frac{9^m}{m}$

Let $S(m) = T(3^m)$

$\therefore S(m) = T(3^m) = 9T(3^{m-1}) + \frac{9^m}{m}$

[redacted]

$S(m) = 9S(m-1) + \frac{9^m}{m}$
Dividing by 9^m throughout,
$\frac{S(m)}{9^m} = \frac{9S(m-1)}{9^m} + \frac{1}{m}$
$\frac{S(m)}{9^m} = \frac{S(m-1)}{9^{m-1}} + \frac{1}{m}$
Let $R(m) = \frac{S(m)}{9^m}$,
$\therefore R(m) = \frac{S(m)}{9^m} = \frac{S(m-1)}{9^{m-1}} + \frac{1}{m}$
$R(m) = R(m-1) + \frac{1}{m}$
$R(m) = \frac{1}{m} + \frac{1}{m-1} + R(m-2)$
(Base case $m = 1, n = 3^1 = 3$ which satisfies our assumption)
$\therefore R(m) = \frac{1}{m} + \frac{1}{m-1} + \frac{1}{m-2} + ... + \frac{1}{2} + \frac{1}{1}$
This is Summation of Harmonic Series which is $\theta(log(m))$ (From Recitation Slides)
$R(m) = \theta(log(m))$
$\implies R(m) = \frac{S(m)}{9^m} = \frac{T(3^m)}{9^m} = \theta(log(m))$
Re-substituting value of m,
$\therefore \frac{T(n)}{n^2} = \theta(log(log_3 n))$
$\implies \mathbf{T(n)} = \theta(\mathbf{n^2 log(log_3 n)})$ as $n- > \infty$

$\square$

[redacted] September 23, 2021

Solutions to Problem 3 of Homework 2 (7+5 points)

[redacted] *Due: 8 am on Thursday, September 23*

Consider the pseudocode for the following randomized algorithm:

```
BLA(n)
    if n ≤ 5 then return 1
    else
        Assign x value of 0 with probability 1/4 and 1 with probability 3/4
        if x = 1 then return BLA(n)
        else return BLA(n/3)
```

(a) (5 points) Let $T(n)$ denote the expected running time of BLA. Derive a recurrence equation for $T(n)$. Solve your recurrence relation using Master Theorem to obtain a asymptotically tight bound.

Solution: If we take T(n) as the Expected running time for the algorithm, then

```
BLA(n)                                                                      ...T(n)
    if n ≤ 5 then return 1                                                  ...θ(1)
    else                                                                    ...θ(1)
        Assign x value of 0 with probability 1/4 and 1 with probability 3/4 ...θ(1)
        if x = 1 then return BLA(n)                                         ...T(n)
        else return BLA(n/3)                                                ...T(n/3)
```

Using the definition of Expectations and Linearity of Expectations,
We can write $T(n) = \frac{3}{4}T(n) + \frac{1}{4}T(\frac{n}{3}) + \theta(1)$
$\therefore \frac{T(n)}{4} = \frac{1}{4}T(\frac{n}{3}) + \theta(1)$
$T(n) = T(\frac{n}{3}) + \theta(1) \qquad \implies a = 1, b = 3, f(n) = \theta(1)$

Calculating $f_{magic}(n) = n^{log_b a} = n^{log_3 1} = n^0 = 1$
We can see that, $f(n) = \theta(f_{magic}(n)) = \theta(1)$ which falls under Case 2 of Master's theorem.
It follows that $T(n) = \theta(f_{magic}(n).\log(n))$
$\therefore T(n) = \theta(n^{log_3 1}.logn)$
$\mathbf{T(n) = \theta(\log(n))}$

□

(b) (2 points) What is the functionality of this algorithm, i.e., what is the expected value returned by this algorithm? Justify your answer briefly.

Solution: We notice that the function is a decreasing function. i.e., the value of n either remains the same or decreases by $(\frac{1}{3})^{rd}$ with a probability of $\frac{1}{4}$ on every recursive call. Hence, we can say that it eventually converges to the Base case which is $= 1$ for $n \leq 5$.
$\implies$ Expected value returned by the algorithm $= 1$.

□

Consider the pseudocode for the following randomized algorithm:

```
FOO(n)
    if n ≤ 1 then return 5
    else
        for i = 1 to n do
            continue
        Assign x value of 0 with probability 1/4 and 1 with probability 3/4
        if x = 1 then return FOO(n)
        else return BLA(FOO(n/3))
```

(c) (**Extra Credit**)(5 points) Let $S(n)$ denote the expected running time of FOO. Derive a recurrence equation for $S(n)$. Solve your recurrence relation by the Recursion Tree Method, by completing the table as shown below:

Solution:

```
FOO(n)                                                                  ..S(n)
    if n ≤ 1 then return 5                                              ...θ(1)
    else
        for i = 1 to n do                                               ...n
            continue                                                    ...θ(1)
        Assign x value of 0 with probability 1/4 and 1 with probability 3/4   ...θ(1)
        if x = 1 then return FOO(n)                                     ...S(n)
        else return BLA(FOO(n/3))                                       ...S(n/3) + θ(logn)
```

where T(n) and S(n) denote the expected running time of $BLA(n)$ and $Foo(n)$, respectively.

Using the definition of Expectations and Linearity of Expectations,
We can write $S(n) = \frac{3}{4}S(n) + \frac{1}{4}(S(\frac{n}{3}) + \theta(logn)) + n + \theta(1)$
$\therefore \frac{S(n)}{4} = \frac{1}{4}(S(\frac{n}{3})) + n + \theta(logn) + \theta(1)$
$S(n) = S(\frac{n}{3}) + 4n + \theta(logn) + \theta(1)$ $\quad (a = 1, b = 3, f(n) = 4n)$

Calculating $f_{magic}(n) = n^{log_b a} = n^{log_3 1} = n^0 = 1$
We can see that, $f(n) >> \theta(f_{magic}(n))$ (as $4n >> \theta(1)$) which falls under Case 3 of Master's

[redacted]

theorem. It follows that $S(n) = \theta(f(n))$
$\implies \mathbf{S(n) = \theta(n)}$

$\square$

[redacted]