H1.PATEL.NISARGS

- Use Latex to prepare your answers. Submit a PDF to Gradescope. Handwritten solutions will not be graded.
- You are permitted to study with friends and discuss the problems; however, *you must write up you own solutions, in your own words*. Do not submit anything you cannot explain. If you collaborate with any of the other students on any problem, please list all your collaborators in your submission for each problem.
- Finding solutions to homework problems on the web, or by asking students not enrolled in the class for answers is not allowed.
- Please use *exactly 1 page* for your answers by placing `\newpage` after your answer. You can delete any portion of the question if you need more space. You will lose points if you ignore this requirement because our grading system depends on it.

PROBLEM 1 *Asymptotic Notation Review*

Rank the following functions by order of growth; that is, write the functions below in order $f_1, f_2, \ldots, f_{12}$ so that $f_i = O(f_{i+1})\ \forall i \in \{1, \ldots, 11\}$. You do not need to provide justification. Hint: use logarithms to simplify the functions.

$$\begin{array}{cccccc} 8^{\sqrt{\log_3 n}} & (\log_2 n)^{(\log_2 n)/(\log_2 \log_2 n)} & n! & n^{1/(\log_9 n)} & \log_2(\log_2(n)) & 1.0005^n \\ 5^{\log_7 n} & \log_2(n!) & n^4 & \sqrt{n} & 2^{\log_5 n} & 2^{(\log_2 n)^5} \end{array}$$

Ans.
$f_1 = n^{1/(\log_9 n)}$
$f_2 = \log_2(\log_2(n))$
$f_3 = 8^{\sqrt{\log_3 n}}$
$f_4 = 2^{\log_5 n}$
$f_5 = \sqrt{n}$
$f_6 = 5^{\log_7 n}$
$f_7 = (\log_2 n)^{(\log_2 n)/(\log_2 \log_2 n)}$
$f_8 = \log_2(n!)$
$f_9 = n^4$
$f_{10} = 2^{(\log_2 n)^5}$
$f_{11} = 1.0005^n$
$f_{12} = n!$
Such that $f_i = O(f_{i+1})\ \forall i \in \{1, \ldots, 11\}$.

PROBLEM 2 *Induction*

In class, we saw an informal argument for why

$$\sum_{i=0}^{n} a^i = \frac{a^{n+1}-1}{a-1}$$

Prove this formula using induction. State a base case, then a hypothesis, and then, that the hypothesis holds for the next larger case.

Ans.

Base case: Let $n = 0$, we have

$$LHS = \sum_{i=0}^{0} a^i = a^0 = 1$$

$$RHS = \frac{a^{0+1}-1}{a-1} = \frac{a-1}{a-1} = 1$$

Let $n = 1$, we have

$$LHS = \sum_{i=0}^{1} a^i = a^0 + a^1 = a + 1$$

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

Thus base case holds.

Hypothesis: Let there be a positive integer k such that the equation is valid for all $n <= k$. Thus,

$$\sum_{i=0}^{k} a^i = \frac{a^{k+1}-1}{a-1}$$

Proof: Now, for $n = k + 1$, we have

$$RHS = \frac{a^{k+2}-1}{a-1}$$

$$LHS = \sum_{i=0}^{k+1} a^i = \sum_{i=0}^{k} a^i + a^{k+1} = \frac{a^{k+1}-1}{a-1} + a^{k+1}$$

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

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

which is the required RHS. Thus by principle of mathematical induction, the property holds for all $n >= 0$.

PROBLEM 3 *Recurrences*

Solve the following recurrences by obtaining a Θ bound. You may assign a standard value for the base case terms $T(1), T(2), \ldots, T(k)$ for some small constant k. Prove your answer. You can use any techniques presented in class.

1. $T(n) = T(n-5) + n$
2. $T(n) = 27T(\lceil n/19 \rceil) + n$
3. $T(n) = 2T(\lceil \sqrt{n} \rceil) + 2$
4. $T(n) = T(\lceil n/11 \rceil) + T(\lfloor 6n/7 \rfloor) + n$

Ans.

1. $\mathbf{T(n) = T(n-5) + n}$

Let $T(k) = 1$ for $0 \leq k < 5$. Thus,

$$\begin{aligned} T(n) &= T(n-5) + n \\ &= (T(n-10) + n) + n \\ &= T(n-10) + 2n \\ &= T(n-15) + 3n \\ &\ldots \\ &= T(n - l * 5) + l * n \end{aligned} \tag{1}$$

The last step would occur when $n - 5l < 5$. Thus for $l = \lceil \frac{n}{5} - 1 \rceil$, and for large n, we have:

$$\begin{aligned} T(n) &= 1 + l * n \\ &= 1 + (\frac{n}{5} - 1)n \\ T(n) &= \frac{n^2}{5} - n + 1 \end{aligned} \tag{2}$$

Thus, we can find 2 values, $c_1 = 0.1$ ans $c_2 = 1$ such that $c_1 n^2 \leq T(n) \leq c_2 n^2$ for large values of n.

Hence,

$$\mathbf{T(n) = \Theta(n^2)}$$

2. $\mathbf{T(n) = 27T(\lceil n/19 \rceil) + n}$

Let $a = 27$, $b = 19$ and $f(n) = n$. Thus we have above equations of the form: $T(n) = aT(n/b) + f(n)$ for large values of n.

We have $n^{\log_b a} = n^{\log_{19} 27} \approx n^{1.12}$. Also we have $f(n) = n = O(n^{1.12-\epsilon})$ for $\epsilon = 0.01$. Hence, by using Case 1 of Master's Theorem, we have:

$$\mathbf{T(n) = \Theta(n^{\log_{19} 27})}$$

3. $\mathbf{T(n) = 2T(\lceil \sqrt{n} \rceil) + 2}$

Substituting, $m = \log(n)$. Thus, $2^m = n$.

Hence we have $T(2^m) = 2T(2^{m/2}) + 2$.

Let $S(m) = T(2^m)$.

Then, $S(m) = 2S(m/2) + 2$.

Let $a = 2$, $b = 2$ and $f(m) = 2$. Thus we have above equations of the form: $S(m) = aS(m/b) + f(m)$ for large values of m.

We have $m^{\log_b a} = m^{\log_2 2} = m$. Also we have $f(m) = 2 = O(m^{1-\epsilon})$ for $\epsilon = 0.01$. Hence, by using Case 1 of Master's Theorem, we have:

$$S(m) = \Theta(m)$$

$$\therefore T(2^m) = \Theta(m)$$

$$\therefore \mathbf{T(n) = \Theta(\log(n))}$$

4. $\mathbf{T(n) = T(\lceil n/11 \rceil) + T(\lfloor 6n/7 \rfloor) + n}$.
Hypothesis: $T(n) = \Theta(n)$

Hypothesis-1: $T(n) \geq n$
Let $T(\lceil n/11 \rceil) = T_1$ and $T(\lfloor 6n/7 \rfloor) = T_2$. Since, T_1 and T_2 will always be ≥ 0, we have $T(n) = T_1 + T_2 + n \geq n$ for all $n \geq 0$. Thus,

$$T(n) = \Omega(n)$$

Hypothesis-2: $T(n) \leq 77n - 100$
Since $T(n) = T(\lceil n/11 \rceil) + T(\lfloor 6n/7 \rfloor) + n$ breaks for $n = 1$. We assume $T(1) = 1$.
Base case:

$$\begin{aligned}
T(2) &= T(\lceil 2/11 \rceil) + T(\lfloor 12/7 \rfloor) + 2 \\
T(2) &= T(1) + T(1) + 2 \\
T(2) &= 1 + 1 + 2 \\
T(2) &= 4 \\
T(2) &\leq 77 * 2 - 100
\end{aligned} \tag{3}$$

Thus base case holds.
Let there be a positive integer k such that the equation is valid for all $n <= k$. Thus,

$$T(k) \leq 77k - 100$$

Proof: Now, for $n = k + 1$, we have

$$\begin{aligned}
T(k+1) &= T(\lceil (k+1)/11 \rceil) + T(\lfloor 6(k+1)/7 \rfloor) + (k+1) \\
&= T((k+1)/11 + 1) + T(6(k+1)/7) + (k+1) \\
&\leq \frac{77(k+12)}{11} - 100 + \frac{6 * 77(k+1)}{7} - 100 + (k+1) \\
&= 74k - 49 \\
&\leq 77k + 77 = 77(k+1)
\end{aligned} \tag{4}$$

which is the required RHS. Thus by principle of mathematical induction, we have $T(n) \leq 77n - 100$. Thus,

$$T(n) = O(n)$$

Since, we have $T(n) = \Omega(n)$ and $T(n) = O(n)$ Hence we can say,

$$\mathbf{T(n) = \Theta(n)}$$

PROBLEM 4 *Master theorem not applicable*

Consider the recurrence $T(n) = 2T(n/2) + f(n)$ in which

$$f(n) = \begin{cases} n^3 & \text{if } \lceil \log(n) \rceil \text{ is even} \\ n^2 & \text{otherwise} \end{cases}$$

Show that $f(n) = \Omega(n^{\log_b(a)+\epsilon})$. Explain why the third case of the Master's theorem does not apply. Prove a Θ-bound for the recurrence.

Ans. We have $a = 1$ and $b = 1$. So, $n^{\log_b(a)+\epsilon} = n^{\log_2(2)+\epsilon} = n^{1+\epsilon}$

$$\begin{aligned} f(n) &= \begin{cases} n^3 & \text{if } \lceil \log(n) \rceil \text{ is even} \\ n^2 & \text{otherwise} \end{cases} \\ &\geq n^2 \ \ \forall \ \ n \geq 0 \\ &\geq n^{1.001} \ \ \forall \ \ n \geq 0 \end{aligned} \tag{5}$$

Thus, $\exists\ \epsilon = 0.001$ and $c = 1$ such that $f(n) \geq c * n^{1+\epsilon}\ \forall\ \ n \geq n_0 = 0$.
Hence $\mathbf{f(n) = \Omega(n^{1+\epsilon})}$.

We have $\log(n/2) = \log(n) - \log(2) = \log(n) - 1$. Thus, if $\log(n)$ is even, then $\log(n/2)$ is odd and vice, versa.

Let for a large value k, $\log(k)$ be even. Thus, $f(k) = k^2$.

Now,

$$af(k/b) = 2f(k/2) = 2(k/2)^3 = k^3/4$$

. We require, $af(n/b) \leq cf(n)$ for some constant $c < 1$. Thus,

$$\begin{aligned} k^3/4 &\leq ck^2 \\ k &\leq 4c \end{aligned} \tag{6}$$

Since $c < 1$. Thus, $k < 4$. Hence $af(n/b) \leq cf(n)$ is not valid for large values of n when $\log(n)$ is even.

Hence third case of Master's theorem cannot be applied for this recurrence relation.

To prove Θ bound for the recurrence. We have

$$\begin{aligned}
T(n) &= 2T(n/2) + f(n) \\
&= f(n) + 2(2(T(n/4) + f(n/2)) \\
&= f(n) + 2f(n/2) + 4T(n/4) \\
&= f(n) + 2f(n/2) + 4f(n/4) + 8T(n/8) \\
&= f(n) + 2f(n/2) + 4f(n/4) + ... + 2^i T(n/2^i) + ... + nT(1)
\end{aligned} \tag{7}$$

We have $T(1) = f(1) = 1$.
Thus, $T(n) = f(n) + 2f(n/2) + 4f(n/4) + ... + 2^i f(n/2^i) + ... + n$
Thus, total we have, $(k+1)$ terms, such that, $2^k = n$ or $k = \log(n)$

Case - 1: $\log(n)$ is even, thus, $f(n) = n^3$. We have,

$$\begin{aligned}
T(n) &= f(n) + 2f(n/2) + 4f(n/4) + ... + 2^i T(n/2^i) + ... \\
&= n^3 + 2(\frac{n}{2})^2 + 4(\frac{n}{4})^3 + 8(\frac{n}{8})^2 + 16(\frac{n}{16})^3 + ... \\
&= n^3 \left(1 + \frac{1}{16} + \frac{1}{16^2} + ...\right) + \frac{n^2}{2}\left(1 + \frac{1}{4} + \frac{1}{4^2} + ...\right)
\end{aligned} \tag{8}$$

Both parts will have $\approx \frac{k}{2} = \frac{\log(n)}{2}$ terms. And using the summation formula mentioned in Q-2, we have

$$\begin{aligned}
T(n) &= n^3 \left(\frac{(\frac{1}{16})^{\frac{\log(n)}{2}+1} - 1}{\frac{1}{16} - 1}\right) + \frac{n^2}{2}\left(\frac{(\frac{1}{4})^{\frac{\log(n)}{2}+1} - 1}{\frac{1}{4} - 1}\right) \\
&= n^3 \left(\frac{1 - (\frac{1}{4})^{\log(n)+2}}{\frac{15}{16}}\right) + \frac{n^2}{2}\left(\frac{1 - (\frac{1}{2})^{\log(n)+2}}{\frac{3}{4}}\right) \\
&= n^3 \left(\frac{16n^2 - 1}{15n^2}\right) + \frac{n^2}{2}\left(\frac{4n - 1}{3n}\right) \\
T(n) &= \frac{16}{15}n^3 + \frac{2}{3}n^2 - \frac{7}{30}n
\end{aligned} \tag{9}$$

Case - 2: $\log(n)$ is odd, thus, $f(n) = n^2$. We have,

$$\begin{aligned}
&= n^2 + 2(\frac{n}{2})^3 + 4(\frac{n}{4})^2 + 8(\frac{n}{8})^3 + 16(\frac{n}{16})^2 + ... \\
&= \frac{n^3}{4}\left(1 + \frac{1}{16} + \frac{1}{16^2} + ...\right) + n^2\left(1 + \frac{1}{4} + \frac{1}{4^2} + ...\right) \\
&= \frac{n^3}{4}\left(\frac{16n^2 - 1}{15n^2}\right) + n^2\left(\frac{4n - 1}{3n}\right) \\
T(n) &= \frac{4}{15}n^3 + \frac{4}{3}n^2 - \frac{7}{20}n
\end{aligned} \tag{10}$$

Thus, in both cases, we have constants $c_1 = 1/15$ and $c_2 = 2$ such that $c_1 * n^3 \leq T(n) \leq c_2 * n^3$ for large values of n. Hence,

$$\mathbf{T(n) = \Theta(n^3)}$$

PROBLEM 5 *Approximate Square Root*

Present and analyze an algorithm that on input $n \in \mathbb{N}$, outputs $\lfloor \sqrt{n} \rfloor$ using $O(\log(n))$ integer ops.

Ans.

SQRT(n):
1. **return** SQRT-HELPER(n, 0, n)

SQRT-HELPER(n, $lower$, $upper$)
1. **if** $upper - lower \leq 1$
2.   **return** lower
3. $mid = \frac{upper - lower}{2}$
4. **if** $mid * mid \leq n$ and $(mid + 1) * (mid + 1) > n$
5.   **return** mid
6. **if** $mid * mid < n$
7.   **return** SQRT-HELPER(n, $mid + 1$, $upper$)
8. **else**
9.   **return** SQRT-HELPER(n, $lower$, $mid - 1$)

The above alogithm uses divide-and-conquer technique to find $\lfloor \sqrt{n} \rfloor$ by continuously dividing the search window in half. SQRT(n) calls the SQRT-HELPER function to provide lower and upper bounds of the answer to be 0 and n respectively.

- In SQRT-HELPER, if condition in Line 1. corresponds to base case when search window is either 0 or 1, thus return the output. It takes 2 operations.
- Line 3 requires 3 operations to get mid.
- Line 4 if statement optimises the code by returning mid if it is the answer. It takes 7 operations.
- Line 5 if statement compares the values of mid*mid and n using 2 operations.
- It calls SQRT-HELPER with half search space either in Line 7 or Line 9, both taking $T(n/2)$ time.

Thus, in worst case, we have, T(n) = T(n/2) + 14.

Let $a = 1$, $b = 2$ and $f(n) = 14$. Thus we have above equations of the form: $T(n) = aT(n/b) + f(n)$ for large values of n.

We have $n^{\log_b a} = n^{\log_2 1} = n^0 = 1$. Also we have $f(n) = 14 = O(1)$. Hence, by using Case 2 of Master's Theorem, we have, $T(n) = \Theta(\log(n))$ in worst case. Thus,

$$\mathbf{T(n) = O(log(n))}$$