22C:44 Homework 3 Solutions

1. (a) $a = 5$, $b = 2$, and $n^{\log_b a} = n^{\log_2 5}$. $f(n) = n^2$. Since $\log_2 5 > 2$ there exists an $\epsilon > 0$ such that $\log_2 5 - \epsilon \geq 2$. Hence $n^2 = O(n^{\log_2 5 - \epsilon})$ for some $\epsilon > 0$. Case (i) of the Master Theorem applies and we get $T(n) = \Theta(n^{\log_2 5})$.

   (b) This solution is almost identical to the above solution except that $f(n) = n^{3/2}$ instead of $f(n) = n^2$. Again Case (i) of the Master Theorem applies and we get $T(n) = \Theta(n^{\log_2 5})$.

   (c) Here $n^{\log_a a} = n$ and $f(n) = n^2 \lg n$. For any ϵ, $0 < \epsilon \leq 1$, $n^2 \lg n = \Omega(n^{1+\epsilon})$. To apply Case (iii) of the Master Theorem we need to check the regularity condition also.

$$af\left(\frac{n}{a}\right) = a\left(\frac{n}{a}\right)^2 \lg\left(\frac{n}{a}\right) = \left(\frac{1}{a}\right) n^2(\lg n - \lg a).$$

   Since $a > 1$, $\lg a \geq 0$ and hence $(\lg n - \lg a) \leq \lg n$. Therefore

$$\left(\frac{1}{a}\right) n^2(\lg n - \lg a) \leq \left(\frac{1}{a}\right) n^2 \lg n.$$

   Since $a > 1$, $1/a < 1$ and therefore the regularity condition holds and by Case (iii) of the Master Theorem we have that $T(n) = \Theta(n^2 \lg n)$.

2. (a) Choose $n_0 = 1$.
   Base Case: We need to show that $T(1) \leq c \cdot 1$. Choosing $c \geq T(1)$ will ensure this.
   Inductive Step: Our *inductive hypothesis* is that for all k, $n_0 \leq k < n$, $T(k) \leq c \cdot k$. We will now show that $T(n) \leq c \cdot n$. Substituting the inductive hypothesis in the recurrence relation for $T(n)$ we get

$$T(n) \leq c\left(\frac{n}{2}\right) + c\left(\frac{n}{3}\right) + n = c\frac{5n}{6} + n.$$

   To show that $T(n) \leq c \cdot n$ it is sufficient to show that $c5n/6 + n \leq cn$ for some positive c. We see that $c5n/6 + n \leq cn$ if $c \geq 6$.
   Hence the base case and the inductive step will work if $c \geq \max\{T(1), 6\}$.

   (b) Choose $n_0 = 2$.
   Base Case: We need to show that $T(2) \geq c \cdot 2\lg(2) = 2c$. Choosing $c \leq T(2)/2$ will ensure this.
   Inductive Step: Our *inductive hypothesis* is that for all k, $n_0 \leq k < n$, $T(k) \geq c \cdot k \lg k$. Substituting the inductive hypothesis into the recurrence relation gives us:

$$\begin{aligned}
T(n) &\geq c\left(\frac{n}{3}\right)\lg\left(\frac{n}{3}\right) + c\left(\frac{2n}{3}\right)\lg\left(\frac{2n}{3}\right) + n \\
&= \frac{cn}{3}[\lg n - \lg 3 + 2\lg 2n - 2\lg 3] + n \\
&= \frac{cn}{3}[\lg n - 3\lg 3 + 2\lg n + 2] + n \\
&= \frac{cn}{3}[3\lg n - 3\lg 3 + 2] + n \\
&= cn\lg n - cn\lg 3 + \frac{2}{3}cn + n
\end{aligned}$$

   To show that $T(n) \geq cn\lg n$, it suffices to show that $cn\lg n - cn\lg 3 + \frac{2}{3}cn + n \geq cn\lg n$. This is equivalent to the condition

$$\frac{2}{3}cn + n \geq cn\lg 3.$$

Solving this for c we get

$$c \leq \frac{1}{\left(\lg 3 - \frac{2}{3}\right)}.$$

Since $\lg 3 > 1 > 2/3$, $1/(\lg 3 - 2/3) > 0$ and hence there is a positive c no greater than this quantity. Choosing $c \leq \min\{T(2), 1/(\lg 3 - 2/3)\}$ will ensure that both the base case and the inductive step go through.

(c) Choose $n_0 = 1$.

Base Case: We need to show that $c_1 \cdot 1 \leq T(1) \leq c_2 \cdot 1$. Choosing $c_1 \leq T(1)$ and $c_2 \geq T(1)$ will ensure this.

Inductive Step: Our *inductive hypothesis* is that for all k, $n_0 \leq k < n$, $c_1 k \leq T(k) \leq c_2 k$. Substituting the inductive hypothesis into the recurrence relation gives us:

$$c_1(\alpha n) + n \leq T(n) \leq c_2(\alpha n) + n.$$

To show that $c_1 n \leq T(n)$, it suffices to show that $c_1 n \leq c_1(\alpha n) + n$. This is true when $c_1 \leq 1/(1-\alpha)$. Similarly, to show that $c_2 n \geq T(n)$, it suffices to show that $c_2 n \geq c_2(\alpha n) + n$. This is true when $c_2 \geq 1/(1-\alpha)$. Since $\alpha < 1$, $1 - \alpha > 0$ and therefore we can choose positive $c_1 \leq \min\{T(1), 1/(1-\alpha)\}$ and $c_2 \geq \max\{T(1), 1/(1-\alpha)\}$.

3.
```
FIND-MISSING(A, n) {
      (* Base Case *)
      if(n == 1)then
            if  (A[1] == 1) then return 0 else return 1;

      ones ← 0; zeroes ← 0;
      for i ← 1 to n do{
            if (LastBit(A[i]) == 0) then{
                  zeroes++; Evens[zeroes] ← RestBits(A[i]);
            }
            if (LastBit(A[i]) == 1) then{
                  ones++; Odds[ones] ← RestBits(A[i]);
            }
      if (zeroes < ⌈(n+1)/2⌉) then
            return Append(FIND-MISSING(Evens, zeroes), 0);
      if (ones < ⌊(n+1)/2⌋) then
            return Append(FIND-MISSING(Odds, ones), 1);
}
```

Explanation: Here it is assumed that `lastBit(A[i])` returns the last bit of element `A[i]`, `RestBits(A[i])` returns the bits of `A[i]` with the last bit removed, and `Append(X, b)` appends the bit `b` to the binary string `X`.

In the for-loop the last bit of each element in `A` is examined and we count the number of elements with 0 as their last bit and the number of elements with 1 as their last bit. These counts, which correspond to the number of even and odd elements respectively, in `A` are compared against what these counts ought to be. Depending on whether an even number is missing or whether an odd number is missing, we recurse either on the set of even numbers or on the set of odd numbers.

Letting $T(n)$ be the running time of this function we see that $T(n) = T(n/2) + \Theta(n)$ for all $n > 1$ and $T(1) = \Theta(1)$. Simplifying this to $T(n) = T(n/2) + n$ does not change the asymptotic value of $T(n)$ and so that is what we use in the Master Theorem. $a = 1$, $b = 2$ and therefore $n^{\log_b a} = n^{\log_2 1} = n^0 = 1$. $f(n) = n$ and therefore we will consider Case (iii) of the Master Theorem. For any ϵ, $0 < \epsilon \leq 1$, $n = \Omega(n^\epsilon)$. The regularity condition $1 \cdot f(n/2) \leq cn$ for some $c < 1$ is trivially satisfied since $f(n/2) = n/2$. Therefore Case (iii) of the Master Theorem applies and $T(n) = \Theta(n)$.