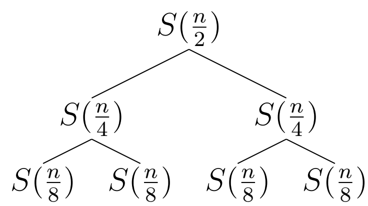
ALGORITHMS, FALL 2018, HOMEWORK 2

Due Thursday, September 20 at 11:59pm.
Worth 2% of the final grade.
Submit each problem on a separate page. Subproblems can be on the same page.

1. $S(n) = 2S(\frac{n}{2}) + \Theta(1)$.

   (a) Evaluate $S(n)$ with a recursion tree.



   Each level of this tree costs $\Theta(1)$, and there are $\log_2 n$ levels.

   $S(1) = \Theta(1) = c_2$

   $S(n) = c_2 n + \log_2 n$

   $S(n) = \Theta(n)$

   (b) Use substitution (induction) to get a lower bound that matches the result in (a).

   $S(n) = 2 \cdot S(\frac{n}{2}) + 1$

   Assume that $S(n) \leq d \cdot n$.

   Therefore for all $k < n$, $S(k) \leq d \cdot k$

   Substitute: $S(n) \leq 2 \cdot d\frac{n}{2} + c \cdot 1$

   $= dn + 1$

   $= dn + 1$

   $\leq dn$ if $d \geq c$

   (c) *Not required or graded: confirm the matching upper bound via substitution.*

   Skipped

2. (a) Use substitution (induction) to prove: $T(n) = 18T(\frac{n}{3}) + \Theta(n^2) = O(n^3)$.

   $T(n) = 18T(\frac{n}{3}) + c \cdot n^2$

   Assume that $T(n) \leq 18 \cdot c \cdot (\frac{n}{3})^2 + d \cdot n^2$

   Therefore for all $k < n$, $T(k) \leq d \cdot k^3$

   Substitute $c \cdot n^3 + dn^2$

   Therefore it is leaf dominated with a runtime of $\Theta(n^3)$.

   (b) Show that this isn't the best possible upper bound for $T(n)$.

   Subsitute: $T(n) \leq c \cdot 2 \cdot n^2 + dn^2$

   $\leq n^2$.

   (c) *Not required or graded: confirm (b) by getting a better bound via substitution.*

3. Use the master method for the following, or explain why it's not possible. If you get Case 3 you do not need to confirm that there is a geometric series.

   (a) $T(n) = 10 \cdot T(\frac{n}{3}) + \Theta(n^2 \log^5 n)$.

   Using master method, we have to compare $n^{\log_3 10}$ and $n^2 \log^5 n$
   $n^{\log_3 10}$ is slightly larger, and it represents the leaves, so this is leaf-dominated.
   Therefore the runtime is $\Theta(n^{\log_3 10})$.

   (b) $T(n) = T(\frac{19n}{72}) + \Theta(n^2)$.

   Using master method, we have to compare $n^0 = 1$ and n^2.
   n^2 is the clear winner here, representing the root cost, so this is root-dominated.
   Therefore the runtime is $\Theta(n^2)$.

   (c) $T(n) = n \cdot T(\frac{n}{5}) + n^{\log_5 n}$.

   Using master method, we have to compare $n^{\log_5 n}$ and $n^{\log_5 n}$.
   Because they're the same, the runtime is $n^{\log_5 n} \log n$.

   (d) $T(n) = 3 \cdot T(\frac{n}{2}) + n^2$.

   Using master method, we have to compare $n^{\log_2 3}$ and n^2.
   n^2 is larger, so this function is root dominated and so the runtime is $\Theta(n^2)$.

   (e) $T(n) = T(\frac{n}{n-1}) + 1$.

   The master method does not work here because the value of b derived from this function would be less than 1.

   (f) $T(n) = 4 \cdot T(\frac{n}{16}) + \sqrt{n} \cdot \log^4 n$.

   Using the master method we have to compare $n^{\log_{16} 4}$ and $\sqrt{n} \cdot \log^4 n$.
   $\sqrt{n} \cdot \log^4 n$ is slightly larger, so this is root-dominated and the runtime is $\Theta(\sqrt{n} \cdot \log^4 n)$.

4. Solve $T(n) = T(\sqrt{n}) + \log n$

(a) with a recursion tree

In this tree there are $\log n$ levels, and on each level we do $\log n$ work, so we have a runtime of $\log(\log n)$.

(b) by substitution (induction)

$T(n) \geq c \cdot \log n$.

For all $k < n, T(k) \geq c \cdot \log k$.

$T(n) \geq c \cdot \log(\sqrt{n}) + \log n$.

$= \frac{1}{2} \cdot c \cdot \log n + \log n$.

Remnants are $c \cdot \log n$.

Therefore this function is $\Theta(\log n)$.

(c) with the master method, after applying a change of variables, $n = 2^m$.

$T(2^m) = T(2^{\frac{m}{2}}) + \log 2^m$

$f(n) = \log 2^m$

If we were to take the entire set of elements as our parameter, and said that $m = n^2$.

Then our equation would equal $S(m) = 4S(\frac{m}{4}) + m$, which is just $\Theta(m)$.

It is similar in the case of n^2, where the runtime would be $\Theta(n^2)$.

Similarly in this one, the runtime just becomes $\Theta(m)$.