10.9
Solving Recurrences

Solving Recurrences

Two general methods:

1. Recursion tree method: need to do sums
   1. elementary methods, geometric series
   2. integration
2. Guess and Verify
   1. guessing involves intuition, experience and trial & error
   2. verification is via induction

Recurrence: Example I

1. Consider $T(n) = 2T(n/2) + n/\log n$ for $n > 2$, $T(2) = 1$.
2. Construct recursion tree, and observe pattern. ith level has 2^i nodes, and problem size at each node is $n/2^i$ and hence work at each node is $\frac{n}{2^i} / \log \frac{n}{2^i}$.
3. Summing over all levels

$$
\begin{aligned}
T(n) &= \sum_{i=0}^{\log n - 1} 2^i \left[\frac{(n/2^i)}{\log(n/2^i)} \right] \\
&= \sum_{i=0}^{\log n - 1} \frac{n}{\log n - i} \\
&= n \sum_{j=1}^{\log n} \frac{1}{j} = nH_{\log n} = \Theta(n \log\log n)
\end{aligned}
$$

Recurrence: Example I

1. Consider $T(n) = 2T(n/2) + n/\log n$ for $n > 2$, $T(2) = 1$.
2. Construct recursion tree, and observe pattern. ith level has 2^i nodes, and problem size at each node is $n/2^i$ and hence work at each node is $\frac{n}{2^i} / \log \frac{n}{2^i}$.
3. Summing over all levels

$$
\begin{aligned}
T(n) &= \sum_{i=0}^{\log n - 1} 2^i \left[\frac{(n/2^i)}{\log(n/2^i)} \right] \\
&= \sum_{i=0}^{\log n - 1} \frac{n}{\log n - i} \\
&= n \sum_{j=1}^{\log n} \frac{1}{j} = nH_{\log n} = \Theta(n \log \log n)
\end{aligned}
$$

Recurrence: Example II

1. Consider $T(n) = T(\sqrt{n}) + 1$ for $n > 2$, $T(2) = 1$.
2. What is the depth of recursion? $\sqrt{n}, \sqrt{\sqrt{n}}, \sqrt{\sqrt{\sqrt{n}}}, \ldots, O(1)$.
3. Number of levels: $n^{2^{-L}} = 2$ means $L = \log\log n$.
4. Number of children at each level is 1, work at each node is 1
5. Thus, $T(n) = \sum_{i=0}^{L} 1 = \Theta(L) = \Theta(\log\log n)$.

Recurrence: Example II

1. Consider $T(n) = T(\sqrt{n}) + 1$ for $n > 2$, $T(2) = 1$.
2. What is the depth of recursion? $\sqrt{n}, \sqrt{\sqrt{n}}, \sqrt{\sqrt{\sqrt{n}}}, \ldots, O(1)$.
3. Number of levels: $n^{2^{-L}} = 2$ means $L = \log\log n$.
4. Number of children at each level is 1, work at each node is 1
5. Thus, $T(n) = \sum_{i=0}^{L} 1 = \Theta(L) = \Theta(\log\log n)$.

Recurrence: Example III

1. Consider $T(n) = \sqrt{n}\,T(\sqrt{n}) + n$ for $n > 2$, $T(2) = 1$.
2. Using recursion trees: number of levels $L = \log\log n$
3. Work at each level? Root is n, next level is $\sqrt{n} \times \sqrt{n} = n$. Can check that each level is n.
4. Thus, $T(n) = \Theta(n \log\log n)$

Recurrence: Example III

1. Consider $T(n) = \sqrt{n}T(\sqrt{n}) + n$ for $n > 2$, $T(2) = 1$.
2. Using recursion trees: number of levels $L = \log\log n$
3. Work at each level? Root is n, next level is $\sqrt{n} \times \sqrt{n} = n$. Can check that each level is n.
4. Thus, $T(n) = \Theta(n \log\log n)$

Recurrence: Example IV

1. Consider $T(n) = T(n/4) + T(3n/4) + n$ for $n > 4$. $T(n) = 1$ for $1 \leq n \leq 4$.
2. Using recursion tree, we observe the tree has leaves at different levels (a lop-sided tree).
3. Total work in any level is at most n. Total work in any level without leaves is exactly n.
4. Highest leaf is at level $\log_4 n$ and lowest leaf is at level $\log_{4/3} n$
5. Thus, $n \log_4 n \leq T(n) \leq n \log_{4/3} n$, which means $T(n) = \Theta(n \log n)$

Recurrence: Example IV

1. Consider $T(n) = T(n/4) + T(3n/4) + n$ for $n > 4$. $T(n) = 1$ for $1 \leq n \leq 4$.
2. Using recursion tree, we observe the tree has leaves at different levels (a lop-sided tree).
3. Total work in any level is at most n. Total work in any level without leaves is exactly n.
4. Highest leaf is at level $\log_4 n$ and lowest leaf is at level $\log_{4/3} n$
5. Thus, $n \log_4 n \leq T(n) \leq n \log_{4/3} n$, which means $T(n) = \Theta(n \log n)$

THE END

...

(for now)