CMSC 351: Recurrence Relations

Justin Wyss-Gallifent

September 28, 2023

1 Introduction

Suppose we are analyzing the time complexity $T(n)$ for an algorithm and suppose that we cannot find an explicit closed formula for $T(n)$ but instead we find a recurrence relation which $T(n)$ satisfies.

A recurrence relation for $T(n)$ tells us how to calculate $T(n)$ for various n in a recursive manner.

Example 1.1. Suppose we have:

$$T(n) = 3T(\lfloor n/5 \rfloor) + 2n + 1 \text{ and } T(1) = 4$$

Then for example here are some easy ones:

$$\begin{aligned} T(1) &= 4 \\ T(5) &= 3T(1) + 2(5) + 1 = 23 \\ T(25) &= 3T(4) + 2(25) + 1 = 120 \end{aligned}$$

A slightly more annoying one is:

$$T(7) = 3T(\lfloor 7/5 \rfloor) + 2(7) + 1 = 3T(1) + 15 = 27$$

An even more annoying one would be $T(2)$ because we're not given enough information. in reality we would need to be given $T(2)$, $T(3)$, and $T(4)$ as well. Luckily since we're only really usually really concerned with large n values we can get away with minimal base cases.

■

Formally a recurrence relation like this should have a floor or ceiling inside the recursive T but in practice this typically dropped for purposes of not getting too tangled up in the calculations.

For example we might just write:

$$T(n) = 3T(n/5) + 2n + 1 \text{ and } T(1) = 4$$

When we do this every specific calculation that follows becomes an approximation when the division yields a non-integer but these approximations are good enough and don't affect time complexity.

There are two goals we might have with a recurrence relation:

1. Find a closed expression for the function, meaning a $T(n) = ...$ which isn't self-referential, or with summations, etc.
2. Find Θ for $T(n)$ if a closed expression is difficult.

2 Solving Using Digging Down

One way to obtain a closed expression is to dig into the recurrence relation. Consider the example:

$$T(n) = 2T(n/2) + \tfrac{1}{2}n \text{ with } T(1) = 7$$

We engage in a process of substitution:

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

But how and when does it end?

Well, the general expression for the above is, for $k = 1, 2, 3, ...$:

$$T(n) = 2^k T\left(\frac{n}{2^k}\right) + \frac{k}{2}n$$

This ends when $n/2^k = 1$ since then we get $T(1) = 7$. This is when $k = \lg n$. At that instant the expression becomes:

$$\begin{aligned}
T(n) &= 2^{\lg n}T(1) + \frac{1}{2}n\lg n \\
&= 7n + \frac{1}{2}n\lg n
\end{aligned}$$

We also see $T(n) = \Theta(n \lg n)$ at this point.

3 Solving Using Trees

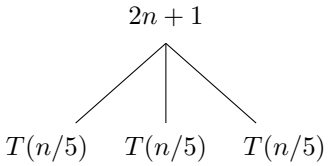
We can also use a recursively generated tree to find an explicit formula for $T(n)$. Such an approach can be messy or not, depending on the recurrence relation

and on the n-values we're analyzing. So as not to go off the deep end, let's consider the example again:

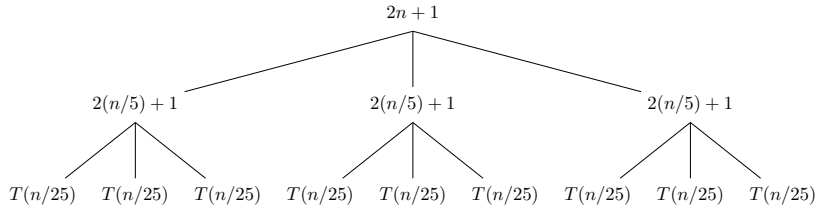
Example 3.1. Suppose $T(n) = 3T(n/5) + (2n + 1)$ with $T(1) = 4$. Let's examine $T(n)$ where $n = 5^k$ for some k. To understand why there's a tree involved we can view the total time done as a very small tree, first with one node:

$$T(n)$$

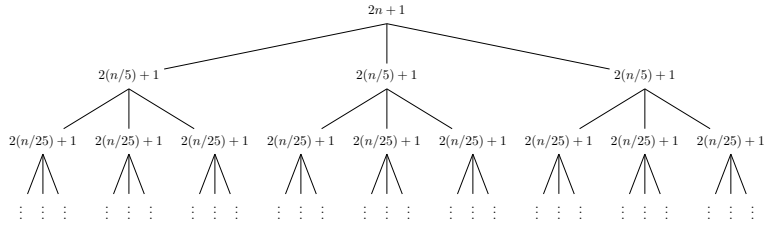
We can expand this according to the recurrence relation, the tree still showing the total time:



Of course each of these leaves is its own problem, the tree still showing the total time.



We could keep going...



But when does it stop?

The tree stops growing when we reach $T(1)$ in the nodes because $T(1) = 4$ and no more recursion happens.

If level $i = 0$ is the top level then as the tree grows we see that level i has entries $T(n/5^i)$ so the leaf level will happen when we reach $T(1)$ which is when $n/5^i = 1$ or $i = \log_5 n = k$. Thus the k^{th} level is the leaf level.

Thus in total there are $k + 1$ levels, 0, 1, ..., k, and in those levels:

Level	Count	Time	Level Time
$i = 0$	1	$2n + 1$	$1(2n + 1) = 3^0 \left(2\left(\frac{n}{5^0}\right) + 1\right)$
$i = 1$	3	$2\left(\frac{n}{5}\right) + 1$	$3\left(2\left(\frac{n}{5}\right) + 1\right) = 3^1 \left(2\left(\frac{n}{5^1}\right) + 1\right)$
$i = 2$	9	$2\left(\frac{n}{25}\right) + 1$	$9\left(2\left(\frac{n}{25}\right) + 1\right) = 3^2 \left(2\left(\frac{n}{5^2}\right) + 1\right)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$i = k - 1$	3^{k-1}	$2\left(\frac{n}{5^{k-1}}\right) + 1$	$3^{k-1}\left(2\left(\frac{n}{5^{k-1}}\right) + 1\right)$
$i = k$	3^k	4	$3^k\,(4)$

Thus the total time is the sum of the levels:

$$
\begin{aligned}
T(n) &= 4(3^k) + \sum_{i=0}^{k-1} 3^i \left(2\left(\frac{n}{5^i}\right) + 1\right) \\
&= 4(3^k) + 2n \sum_{i=0}^{k-1} \left(\frac{3}{5}\right)^i + \sum_{i=0}^{k-1} 3^i \\
&= 4(3^k) + 2n \left(\frac{1 - \left(\frac{3}{5}\right)^k}{1 - \frac{3}{5}}\right) + \left(\frac{3^k - 1}{3 - 1}\right) \\
&= 4(3^k) + 2n \left(\frac{5}{2}\right) \left(1 - \left(\frac{3}{5}\right)^k\right) + \frac{1}{2}\left(3^k - 1\right) \\
&= 4(3^{\log_5 n}) + 5n - 5n \left(\frac{3}{5}\right)^{\log_5 n} + \frac{1}{2}\left(3^{\log_5 n}\right) - \frac{1}{2} \\
&= \frac{9}{2}(3^{\log_5 n}) + 5n - 5n \left(\frac{3^{\log_5 n}}{5^{\log_5 n}}\right) - \frac{1}{2} \\
&= \frac{9}{2}(3^{\log_5 n}) + 5n - 5\left(3^{\log_5 n}\right) - \frac{1}{2} \\
&= -\frac{1}{2}(3^{\log_5 n}) + 5n - \frac{1}{2} \\
&= 5n - \frac{1}{2}(3^{\log_5 n} + 1)
\end{aligned}
$$

Note that results from this equation will agree with the calculations we did earlier. For example:

$$T(5) = 5(5) - \frac{1}{2}(3^{\log_5 5} + 1) = 25 - \frac{1}{2}(3 + 1) = 23$$

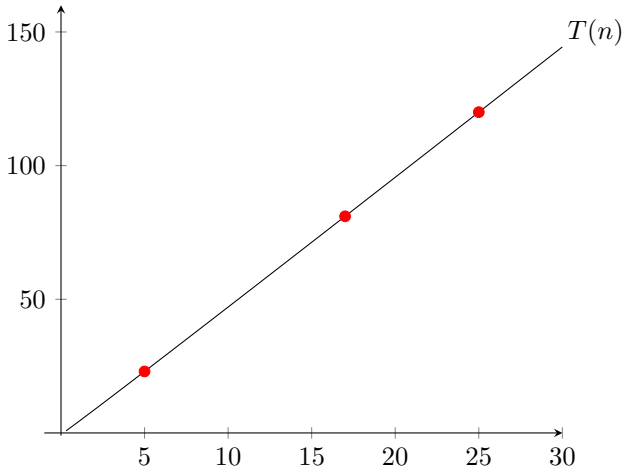
And:

$$T(25) = 5(25) - \frac{1}{2}(3^{\log_5 25} + 1) = 125 - \frac{1}{2}(9 + 1) = 120$$

While this formula is exact only for $n = 5^k$ it gives us a good approximation in other cases:

$$T(17) = 5(17) - \frac{1}{2}(3^{\log_5 17} + 1) \approx 81.04$$

Here is a graph of $T(n)$ as well as these points:



As far as time complexity observe that:

$$T(n) = 5n - \frac{1}{2}(3^{\log_5 n} + 1) < 5n$$

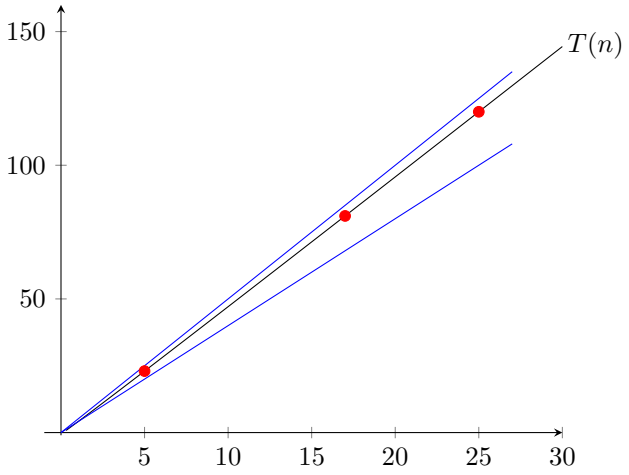
Thus $T(n) = \mathcal{O}(n)$.

And observe that since $\log_5 n < \log_3 n$ we have $3^{\log_5 n} < 3^{\log_3 n} = n$ so that for $n \geq 1$ we have:

$$T(n) = 5n - \frac{1}{2}(3^{\log_5 n} + 1) > 5n - \frac{1}{2}(n + 1) = \frac{9}{2}n - \frac{1}{2} \geq 4n$$

Thus $T(n) = \Omega(n)$ and together we have $T(n) = \Theta(n)$.

This corresponds to the picture in the sense that it certainly appears that $Bn \leq T(n) \leq Cn$ for some $B, C > 0$ and for sufficiently large n. In fact here is the same plot but with $4n$ and $5n$ plotted as well:



■

By popular demand here is a second tree example but more streamlined.

Example 3.2. Consider the recurrence relation given here:

$$T(n) = 2T(n/2) + \sqrt{n} \text{ with } T(1) = 3$$

For the sake of simplicity assume $n = 2^k$ for some k. During the growth of the tree the nodes in level i are $T(n/2^i)$ before being replaced by $\sqrt{n/2^i}$ as the tree grows down. The leaf level occurs when $n/2^i = 1$ or $i = \lg n = k$.

Thus in total there are $k+1$ levels, 0, 1, ..., $k+1$ and in those levels:

Level	Count	Time	Total Time
$i = 0$	1	$\sqrt{n}$	$1\sqrt{n} = \sqrt{1n}$
$i = 1$	2	$\sqrt{n/2}$	$2\sqrt{n/2} = \sqrt{2n}$
$i = 1$	4	$\sqrt{n/4}$	$4\sqrt{n/4} = \sqrt{4n}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$i = k-1$	2^{k-1}	$\sqrt{n/2^{k-1}}$	$2^{k-1}\sqrt{n/2^{k-1}} = \sqrt{2^{k-1}n}$
$i = k$	2^k	3	$2^k\,(3)$

Thus the total time is:

$$\begin{aligned}
T(n) &= 3\left(2^k\right) + \sum_{i=0}^{k-1} \sqrt{2^i n} \\
&= 3n + \sqrt{n} \sum_{i=0}^{k-1} (2^i)^{1/2} \\
&= 3n + \sqrt{n} \sum_{i=0}^{k-1} (2^{1/2})^i \\
&= 3n + \sqrt{n} \left(\frac{(2^{1/2})^k - 1}{2^{1/2} - 1} \right) \\
&= 3n + \sqrt{n} \left(\frac{(2^k)^{1/2} - 1}{2^{1/2} - 1} \right) \\
&= 3n + \sqrt{n} \left(\frac{\sqrt{n} - 1}{2^{1/2} - 1} \right) \\
&= 3n + \frac{1}{\sqrt{2} - 1}(n - \sqrt{n}) \\
&= 3n + (\sqrt{2} + 1)(n - \sqrt{n})
\end{aligned}$$

This checks with the recurrence relation since for example the recurence relation gives us $T(2) = 2T(1) + \sqrt{2} = 6 + \sqrt{2}$ and this formula gives us

$T(2) = 3(2) + (\sqrt{2}+1)(2-\sqrt{2}) = 6+\sqrt{2}$ and for example the recurrence relation gives us $T(4) = 2T(2) + \sqrt{4} = 2(6+\sqrt{2}) + 2 = 14 + 2\sqrt{2}$ and this formula gives us $T(4) = 3(4) + (\sqrt{2}+1)(4-\sqrt{4}) = 14 + 2\sqrt{2}$.

■

4 Thoughts, Problems, Ideas

1. For the example $T(n) = 3T(n/5) + (2n + 1)$ with $T(1) = 4$ show that calculating $T(125)$ directly (using the recurrence relation) and using the formula developed in the notes yields the same results.
2. For the example $T(n) = 2T(n/5) + (2n + 3)$ with $T(1) = 4$ draw the complete tree for $n = 125$ and fill in the values. What is the total time?
3. Suppose $T(n) = 2T(n/5) + (2n + 1)$ and $T(1) = 2$. Emulate the example in the notes in the following sense:
   (a) Calculate $T(n)$ for a few values which are nice powers (of what?)
   (b) Draw a generic version of the associated tree.
   (c) Calculate the number of levels in the tree.
   (d) Calculate the number of entries in each level.
   (e) Separately for each non-leaf level calculate the total time in each entry and then add these to get the total time each non-leaf level.
   (f) Calculate the total time in the leaf-level.
   (g) Write down a sum for the total time in the tree.
   (h) Simplify this sum to get the total time $T(n)$.
   (i) Use this $T(n)$ to check your values from (a).
   (j) Calculate the $\mathcal{O}(n)$ time complexity from $T(n)$.
   (k) Calculate the time complexity from the Master Theorem.
   (l) Rejoice at the beauty of equality.
4. Repeat the previous problem with $T(n) = 2T(n/4) + \sqrt{n}$ and $T(1) = 4$.
5. Repeat the previous problem with $T(n) = 3T(n/3) + 2$ and $T(1) = 7$.