Topics

1. Recurrences (cont.)
    - Substitution Method
    - Recursion Trees
2. Combinatorics and Permutations

1 Recurrences

1.1 Substitution Method

The idea of the substitution method is to guess a solution and then prove it by induction.

Example

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

Guess: $T(n) = \mathcal{O}(n \lg n)$

Induction Hypothesis: For all $k < n$, $T(k) \leq dk \lg k$

Induction:

$$\begin{aligned} T(n) &= 2T\left(\frac{n}{2}\right) + \Theta(n) \\ T(n) &\leq 2d\frac{n}{2} \lg\left(\frac{n}{2}\right) + cn \\ &= dn \lg n - dn \lg 2 + cn \\ &= dn \lg n - dn + cn \end{aligned}$$

Which implies that $T(n) \leq dn \lg n$ if:

$$\begin{aligned} cn - dn &\leq 0 \\ cn &\leq dn \\ c &\leq d \end{aligned}$$

This proves $T(n) = \mathcal{O}(n \lg n)$. We could also prove the lower bound similarly.

What about the base cases for the induction? We had $T(1) = 1$, but $dn \lg n = 0$ when $n = 1$, so the base case was not satisfied at $n = 1$. Generally, we hand-wave away the base cases. We are interested in asymptotic behavior (that is, for large n). If the induction holds and the base of the recurrence is $\mathcal{O}(1)$, it will be the case that there is a basis step that works. We generally don't try to find it. How would we even do so in this example, since there is a $\Theta(n)$ term for which we do not know the constant? Note that if instead we had $f(n) = n$, we could easily find a base case that holds.

Another very important note is that we must prove the exact form of the induction hypothesis. Let's look at an incorrect proof to see why.

Example

Consider the same recurrence $T(n) = 2T(\frac{n}{2}) + \Theta(n)$. This time let's guess incorrectly that it is $\mathcal{O}(n)$.

Hypothesis: $T(k) \leq dk$ for all $k < n$

Induction:

$$\begin{aligned} T(n) &= 2T\left(\frac{n}{2}\right) + \Theta(n) \\ T(n) &\leq 2d\frac{n}{2} + cn \\ &= dn + cn \\ &= \cancel{\mathcal{O}(n) \text{ so we're done}} \end{aligned}$$

We must prove the exact form of the hypothesis. That is, $T(n) \leq dn$. However, $dn + cn$ is not less than or equal to dn for any $c > 0$. Therefore, we have *not* proven that $T(n) = \mathcal{O}(n)$ (and indeed we could not prove that as it is false).

A key part of the substitution method is making guesses. But how do we guess?

- See if we know the answer to a similar recurrence
- Guess a loose upper bound and lower bound, and then gradually refine each until they are equal.
- Change of variables, as we saw in the Master Theorem section
- Recursion trees

1.2 Recursion Trees

A recursion tree is a way of visualizing a divide-and-conquer computation where each node represents a subproblem. We can be sloppy when we're drawing them, the point is to get a reasonable guess, not to provide a proof. A recursion tree is *not* a proof, and will not be accepted as such in homeworks or exams.

Example

$T(n) = 3T(\frac{n}{4}) + n^2$, $T(1) = \Theta(1)$

Draw a few levels of the recursion tree (3 is good). First, let's examine how the size of subproblems decreases. At each level, it goes down by a factor of 4. Therefore, at depth i we're dealing with subproblems of size $\frac{n}{4^i}$. The size reaches 1 (*i.e.*, the boundary condition) when $\frac{n}{4^i} = 1$. In other words, our height is going to be about:

$$h = \mathcal{O}(\log_4 n)$$

Now let's look at the work done at each level. At the root, it's n^2. At the next level, $\frac{3}{16}n^2$. At the next level, $\left(\frac{3}{16}\right)^2 n^2$. A clear pattern is emerging where the work at level i is

$$\left(\frac{3}{16}\right)^i n^2$$

The total amount of work done in the whole tree is something like:

$$\begin{aligned}
\sum_{i=0}^{\log_4 n} \left(\frac{3}{16}\right)^i n^2 &< \sum_{i=0}^{\infty} \left(\frac{3}{16}\right)^i n^2 \text{ (bounding by an infinite extension of the series)} \\
&= \frac{1}{1-\frac{3}{16}} n^2 \text{ (sum of infinite geometric series)} \\
&= \frac{16}{13} n^2 = \mathcal{O}(n^2)
\end{aligned}$$

However, we've neglected the boundary conditions (*i.e.*, the places where $T(1)$ appears). There are $n^{\log_4 3}$ leaf nodes, each contributing $T(1) = \Theta(1)$. Therefore, these contribute $\Theta(n^{\log_4 3})$. This is $\mathcal{O}(n^2)$, so the n^2 will still dominate.

So, we would guess $\mathcal{O}(n^2)$. Why not $\mathcal{O}(n^2 \lg n)$? We could have guessed that using less precise reasoning that each level does $\mathcal{O}(n^2)$ work and there are $\mathcal{O}(\lg n)$ levels. However, by computing the sum above, we were able to come up with a tighter bound. Now, let's prove it with substitution.

Hypothesis: $T(k) \leq ck^2 \quad \forall k < n$

Induction:

$$\begin{aligned}
T(n) &= 3T\left(\frac{n}{4}\right) + n^2 \\
&\leq 3c\left(\frac{n}{4}\right)^2 + n^2 \\
&= \frac{3cn^2}{16} + n^2 \\
&= n^2\left(\frac{3c}{16} + 1\right) \\
&\leq cn^2
\end{aligned}$$

Where the last step holds if:

$$\begin{aligned}
\frac{3c}{16} + 1 &\leq c \\
3c + 16 &\leq 16c \\
c &\geq \frac{16}{13}
\end{aligned}$$

So, $T(n) = \mathcal{O}(n^2)$. We could prove the lower Ω bound similarly. But, we could do something even easier: it's very obvious that $T(n) = \Omega(n^2)$ because $T(n)$ equals n^2 plus the recursive term, which is definitely non-negative. Therefore $T(n) = \Theta(n^2)$.

2 Combinatorics and Permutations

Rule of Sum: The number of ways to choose one element from one of two disjoint sets A and B is the sum of their cardinalities $|A| + |B|$.

Rule of Product: The number of ways to choose an ordered pair is the number of ways to choose the first times the number of ways to choose the second. For instance, if $|A|$ and $|B|$ are sets and we want to choose one element from A and one element from B, there are $|A| \times |B|$ ways to do so.

Permutation: A permutation of a set S is an ordered sequence of all elements of S with each element appearing exactly once. A k-permutation of S is a permutation of a k-subset of S. The number of k permutations of an n-set are:

$$\frac{n!}{(n-k)!}$$

Why? We have n ways to choose the first element, $(n-1)$ for the second, and so on down to $(n-k+1)$. This gives $n(n-1)\dots(n-k+1)$.

Combination: A k-combination of a set S is simply a k-subset of S. The number of k-combinations of an n-set is equal to:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Note that we can get the same result by taking the number of k-permutations and dividing by $k!$. Why is that? Each k-combination can be ordered $k!$ different ways, meaning it corresponds to $k!$ different k-permutations, and so is double-counted $k!$ times by the permutation formula.

Exercise C.1-4

How many ways can we choose three numbers from $\{1, \dots, 99\}$ so that their sum is even?

We need to have either three even numbers or one even and two odd numbers. There are 49 even numbers and 50 odd ones to choose from. Therefore, the answer is

$$\binom{49}{3} + 49 \times \binom{50}{2} = 78449$$

Exercise C.1-9

Prove the following using a combinatorial argument:

$$\sum_{i=1}^{n} i = \binom{n+1}{2}$$

This is easy to prove algebraically; it's the well-known formula for the sum of the first n positive integers.

Combinatorially, we argue as follows. Consider a group of $n+1$ people, where everyone shakes hands with everyone else. The right hand side counts the number of handshakes directly (we choose any 2 people from $n+1$ and they shake hands).

The left-hand side counts the same thing differently. Imagine lining up the $n+1$ people and numbering them 1 to $n+1$. Each person then shakes hands with everyone on their right. This, person 1 initiates n handshakes, person 2 initiates $n-1$, ..., and person $n+1$ initiates no handshakes. This is exactly the sum on the left-hand side.

Why couldn't we just say there are $n(n+1)$ pairs of people, so that's how many handshakes there are? That would double-count each handshake, so we'd need to divide by two. Note that $\frac{n(n+1)}{2} = \binom{n+1}{2}$, which is exactly the right-hand side.