CSE 241 Class 5

Jeremy Buhler

September 9, 2015

Today: **the Master Method**

1 Recursion Trees Suck

- **We're all pretty tired of recursion trees.**
- Many recursions can be solved by same basic process.
- Can we do one recursion tree analysis that gives a general solution to a whole class of recurrences?

Let's consider the general recurrence:

$$T(n) = \begin{cases} c_0 & \text{if } n = 1 \\ aT(n/b) + f(n) & \text{otherwise} \end{cases} .$$

What do we know about solving this recurrence? Recursion tree, of course. Let's apply four-step solution procedure. *Step 1*: draw it (assume n is power of b).

We've now accounted the cost of the algorithm. Let's write down the summation:

$$T(n) = \sum_{k=0}^{\log_b n-1} a^k f(n/b^k) + c_0 n^{\log_b a}$$

Remember this summation!

2 A Comment on Asymptopia

Suppose we stop the recurrence for some constant $n_0 > 1$? How would this change alter the summation?

- Not immediately obvious. Needs some algebra.
- Answer, however, is straightforward: time becomes

$$\begin{aligned} T'(n) &= T(n) - c^* n^{\log_b a} \\ &= \sum_{k=0}^{\log_b n-1} a^k f(n/b^k) + (c_0 - c^*) n^{\log_b a} \end{aligned}$$

- **This change cannot affect the asymptotic behavior of the recurrence!**

In practice, we can ignore base cases when discussing asymptotic solutions of recurrences. For asymptotic analysis, can write:

$$T(n) = aT(n/b) + f(n)$$

without specifying base case.

3 Qualitative Behavior of General Recurrence

What is asymptotic solution to recurrence for $T(n)$? Let's rewrite the exact solution a little bit...

$$T(n) = f(n) + \sum_{k=1}^{\log_b n-1} a^k f(n/b^k) + c_0 n^{\log_b a}$$

We will consider three cases for large n:

1. Third term dominates. Then $T(n) = \Theta(n^{\log_b a})$. (Base case work dominates.)
2. First and third terms balance; that is, $f(n) = cn^{\log_b a}$. Each level of tree takes time $a^k n^{\log_b a}/b^{k \log_b a}$, which is $(a^k/a^k)n^{\log_b a}$. Hence, $T(n) = \Theta(f(n) \log n)$.
3. First term dominates. Then $T(n) = \Theta(f(n))$. (Top-level work dominates, recursion is cheap.)

This only gives intuition for what's going on. We'll give a precise theorem next.

4 Statement of The Master Method

Theorem: consider a recurrence of the form

$$T(n) = aT(n/b) + f(n)$$

where $a \geq 1$, $b > 1$ are constants and $f(n)$ is a non-negative function of n.

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a} \log^k n)$, $k \geq 0$, then $T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$.
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for $\epsilon > 0$ and "f is not strange," then $T(n) = \Theta(f(n))$.

"f is not strange": a *regularity condition* (technical point needed by the proof). Specifically,

For some $c' < 1$ and all large enough n,

$$af(n/b) \leq c' f(n)$$

This condition is quite difficult to violate in practice but must be given for completeness.

- **Note 1**: In cases 1 and 3, there must be a *polynomial* gap between $f(n)$ and $n^{\log_b a}$. I.e., these three cases are *not exhaustive*.
- **Note 2**: This theorem holds even if the original recurrence contains floors and ceilings. That is,

  $$T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + f(n)$$

  is treated the same as

  $$T(n) = 2T(n/2) + f(n).$$

5 Examples Using the Master Method

Let's do some examples.

- $$T(n) = 3T(n/4) + 7n$$

  1. What are a and b? $a = 3$, $b = 4$.
  2. Observe that $\log_4 3 \approx 0.79$.
  3. n^1 is polynomially [**wait**] greater than $n^{\log_4 3}$.
  4. Case 3: solution is $\Theta(n)$.

- $$T(n) = 5T(n/3) + 7n$$

  1. What are a and b? $a = 5$, $b = 3$.

2. Observe that $\log_3 5 \approx 1.46$.
3. n^1 is polynomially [**wait**] less than $n^{\log_3 5}$.
4. Case 1: solution is $\Theta(n^{\log_3 5})$.

-
$$T(n) = 5T(n/3) + 7n \log n \log \log n$$

  1. What are a and b? $a = 5$, $b = 3$.
  2. Observe that $\log_3 5 \approx 1.46$.
  3. $n^1 \log n \log \log n$ is polynomially [**wait**] less than $n^{\log_3 5}$.
  4. Case 1: solution is $\Theta(n^{\log_3 5})$.

-
$$T(n) = 2T(n/2) + cn$$

  1. What are a and b? $a = 2$, $b = 2$.
  2. Observe that $\log_2 2 = 1$.
  3. n^1 is [**wait**] the same as n^1.
  4. Case 2: solution is $\Theta(n \log n)$.

-
$$T(n) = 2T(n/2) + cn \log n$$

  1. What are a and b? $a = 2$, $b = 2$.
  2. Observe that $\log_2 2 = 1$.
  3. $n^1 \log n$ is [**wait**] polynomially the same order as n^1.
  4. Case 2: solution is $\Theta(n \log^2 n)$.

-
$$T(n) = 4T(n/2) + cn^2$$

  1. What are a and b? $a = 4$, $b = 2$.
  2. Observe that $\log_2 4 = 2$.
  3. n^2 is [**wait**] the same as n^2.
  4. Case 2: solution is $\Theta(n^2 \log n)$.

-
$$T(n) = 4T(n/2) + cn^2/\log n$$

  1. What are a and b? $a = 4$, $b = 2$.
  2. Observe that $\log_2 4 = 2$.
  3. $n^2/\log n$ is [**wait**] polynomially of same order as n^2, but difference is not a multiplicative log factor.
  4. *No answer*: falls into gap between cases.

What do we do in this case? Recursion tree (as on homework), informed guess, etc.