6.046 Recitation 2: Recurrences Bill Thies, Fall 2004

Acknowledgement

Parts of these notes are adapted from notes of George Savvides and Dimitris Mitsouras.

My contact information:

Bill Thies

thies@mit.edu

Office hours Wed. 7-9pm, 36-153 (this room)

Announcements:

PS 1 due Mon 9/20

PS 2 out Mon, due Wed 9/29

Today: Recurrences

- Master theorem
- Tricks
- Substitution
- Sloppiness

Master Theorem

Should memorize all 3 cases. They are surprisingly simple if you forget the fine print:

Given recurrence $T(n) = aT(n/b) + f(n)$
$A \geq 1$, $b > 1$, $f(n)$ asym. positive

A) Compute $n^{\log_b a}$ and compare that with $f(n)$.
B) If there's a clear winner, $T(n) = \Theta(\text{WINNER})$
C) If they're both the same order, $T(n) = \Theta(n^{\log_b a} \log n)$ [add a log]

In detail:

1. $f(n) = O(n^{\log_b a} - \varepsilon)$ with $\varepsilon > 0$ ==> $T(n) = \Theta(n^{\log_b a})$
2. $f(n) = \Theta(n^{\log_b a} \log^k n)$ with $k \geq 0$ ==> $T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$
3. $f(n) = \Omega(n^{\log_b a} + \varepsilon)$, $\varepsilon > 0$ and reg. condition ==> $T(n) = \Theta(f(n))$
   regularity condition: $a\, f(n/b) \leq c\, f(n)$ with $c<1$ forall $n > n_0$

Intuition again:
case 1: the leaves dominating
case 2: leaves and combining equal
case 3: root dominating
  * regularity condition shows that total work decreases as you go down
  - almost everything will satisfy the regularity condition
    - SATISFIES: n^k, $\lg^k(n)$, 2^k, $n!$
    - DOES NOT SATISFY: later today

Master Theorem Examples

Case 1: Strassen T(n)

$7T(n/2) + \Theta(n^2)$

$T(n) = \Theta(n^{\log_2 7})$, about $\Theta(n^{2.8})$

Case 2: Merge Sort

$T(n) = 2T(n/2) + \Theta(n)$

$T(n) = \Theta(n \log n)$

Case 3: $T(n) = T(n/2) + n$

$T(n) = \Theta(n)$

Case that doesn't work:

$T(n) = 2T(n/2) + n \lg \lg n$

None of the cases are satisfied; falls in between 2 and 3.

(Note that $\lg^k n = (\lg n)^k$. That is different than $\lg \lg n$).

For many more examples + solutions, see handout.

What doesn't satisfy the regularity condition?

Idea: you want f(n) to possibly grow (or shrink by only a factor of a) when you decrease n to n/b.

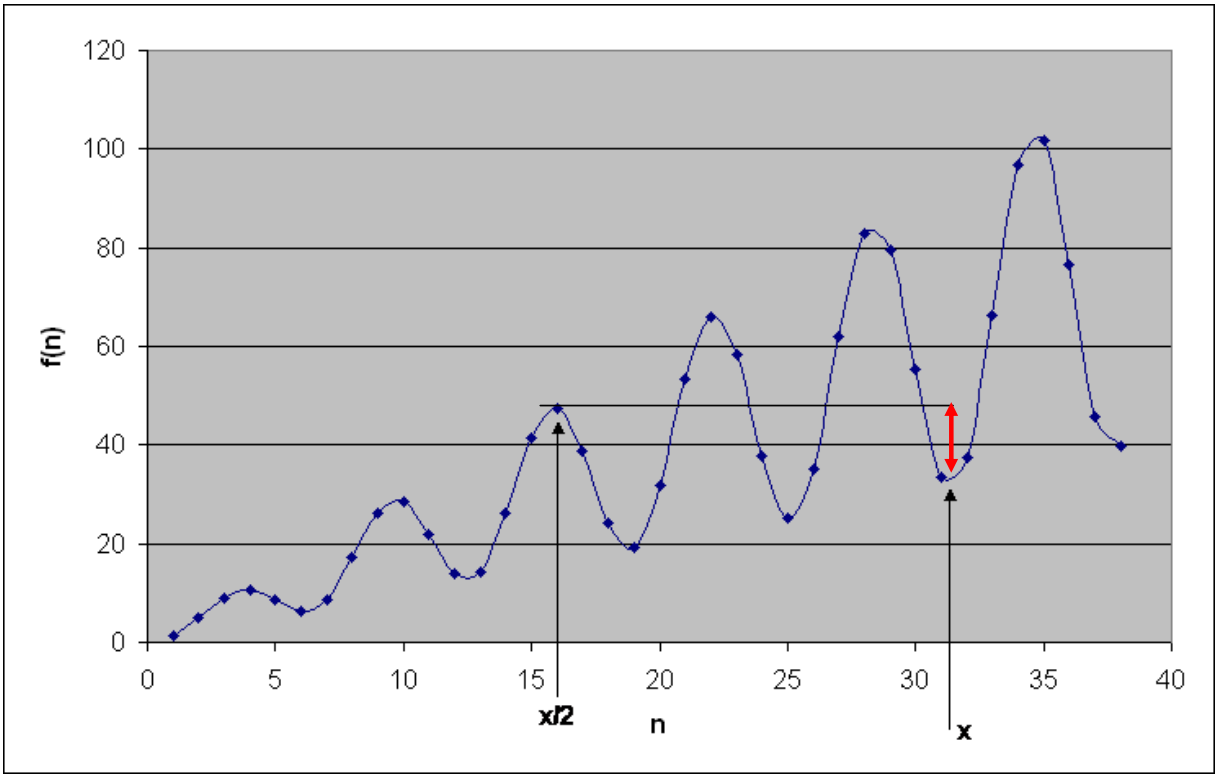
Well, functions like 1/n and $(0.5)^n$ would not be regular, but with these values of f you couldn't choose values of a and b so that Case 3 was invoked in the first place. We need an f(n) that is $\Omega(n^k)$ for some $k > 0$.

It is actually a little bit tricky to find a case where we select Case 3 but then find that the regularity condition is violated. Virtually all functions you'll see in this class will satisfy the regularity condition.

Here is one case where the regularity condition makes a difference:

$T(n) = T(n/2) + n(\sin(n - \pi/2) + 2)$

Have a look at this $f(n) = n(\sin(n-\pi/2) + 2)$. The interesting thing is that you can choose x s.t. f(x/2) is bigger than f(x), even though $f(n) \geq n$ forall n.



What doesn't satisfy the regularity condition?

Here's the formal analysis:

$T(n) = T(n/2) + n(\sin(n - \pi/2) + 2)$

We are in Case 3 of the Master Theorem, and the regularity condition is:

$(n/2)\,(\sin(n/2 - \pi/2) + 2) <= c\, n(\sin(n - \pi/2) + 2)$

$$c >= (1/2)\ \frac{\sin(n/2 - \pi/2) + 2}{\sin(n - \pi/2) + 2}$$

To see it is impossible to satisfy this for all large n, choose:

$n = 2\,\pi\,k$

where k is odd.

\|/ relies on k being odd

Then $\sin(n/2 - \pi/2) = \sin(\pi\,k - \pi/2) = \sin(\pi/2) = 1$

$\sin(n - \pi/2) = \sin(2\,\pi\,k - \pi/2) = \sin(-\pi/2) = -1$

And we get

$$c >= (1/2)\ \frac{1 + 2}{-1 + 2}$$

$c >= 3/2$

Thus, we can't choose $c < 1$ to satisfy the condition.

Trick 1: Change of variable

A nasty-looking recurrence can look more familiar if we change variables.

$T(n) = 2T(\text{root}(n)) + \lg(n)$ → Master Theorem does not apply, because argument to T(n) is not T(n/b) for const. b

let $m = \lg(n) \Rightarrow n = 2^m$

$T(2^m) = 2T(\text{root}(2^m)) + \lg(2^m)$
$= 2T(2^{m/2}) + m$

let $S(m) = T(2^m)$

$S(m) = 2S(m/2) + m$

Solve recurrence: $S(m) = O(m \lg m)$

Now get T(n) again:

$T(n) = T(2^m)$
$= S(m)$
$= O(m \lg m)$
$= O(\lg n \lg \lg n)$

Trick 2: Ignoring lower-order terms when guessing a bound

$T(n) = 2T(n/2 + \text{root}(n)) + \Theta(1)$ → Master Theorem does not apply.

for large n, n/2 >> root n
root(n) = o(n/2)

So intuitively it should be equivalent to:

$S(n) = 2S(n/2) + \Theta(1)$

Solve recurrence using Master Theorem: $T(n) = \Theta(n)$

This is only a guess, not a proof! Need to check resulting bound using substitution method.

Substitution method

Consider example: $T(n) = T(n/3) + T(2n/3) + n$

First, draw recursion tree to estimate bound on T(n):

Step 1:

n

T(n/3) T(2n/3)

Step 2: Work per level

n .....n

n/3 2n/3 .....n

T(n/9) T(2n/9) T(2n/9) T(4n/9).....n

} Height of tree: $\log_{3/2}(n)$

Total: O(n lg n)

- This is only a guess, need to verify with substitution method:

Want to show: $T(n) = O(n \lg n)$

Inductive hypothesis: $T(n) \leq c\, n \lg n$

Base case: we know is constant time

Inductive step:

$T(n) = T(n/3) + T(2n/3) + n$

$\leq c\,(n/3) \lg (n/3) + c\,(2n/3) \lg (2n/3) + n$

$= c\,(n/3)(\lg n - \lg 3) + c\,(2n/3)(\lg 2 + \lg n - \lg 3) + n$

$= c\, n \lg n - c\, n \lg 3 - c\,(2n/3) \lg 2 + n$

$= \underbrace{c\, n \lg n}_{\textit{base}} - \underbrace{n\,(c \lg 3 + c(2/3) \lg 2 - 1)}_{\textit{residual}}$

Residual is > 0, e.g. for c=1

$\leq c\, n \lg n$

Sloppiness

We didn't have time to cover this in recitation, but it's **cool!**

Idea: if we can solve T(n) for $n=2^k$, we can usually extend bound
to all n>0

Sloppiness theorem:

- Given T(n) and f(n), then T(n) = O(f(n)) if these conditions hold:

1) forall k>0, $T(n) \leq c\, f(2^k)$ [works on powers of 2]

2) T(n) and f(n) are monotonically increasing
$T(n) \leq T(n+1)$, $f(n) \leq f(n+1)$ forall n>0

3) f(n) is "slowly growing", i.e. f(n) = O(f(n/2))
TRUE if $f(n) = n^k$, or $\lg^k n$
FALSE if $f(n) = k^n$

Theorem holds for any b>1, not just 2. (Presenting it as 2 to simplify.)

Proof of Sloppiness Theorem

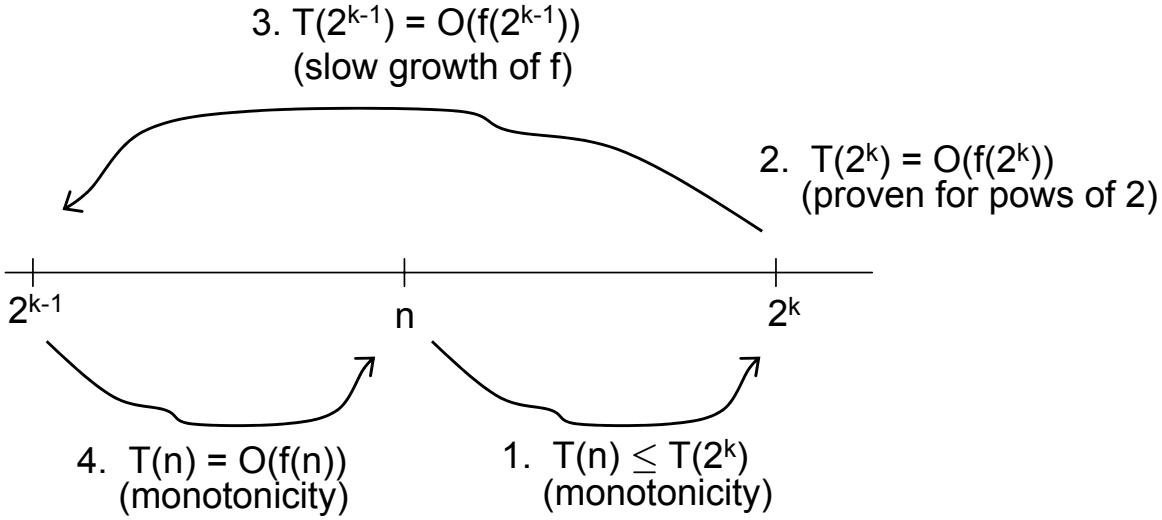
Goal: $T(n) = O(f(n)) \; \forall \; n > n_0$

Proof:

$T(n) = T(2^{\lg n})$		[math. Identity]
$\leq T(2^{\lceil \lg n \rceil})$		[monotonicity (condition #2)]
$\leq c\, f(2^{\lceil \lg n \rceil})$		[works for powers of 2 (condition #1)]
$\leq c\, d\, f(2^{\lceil \lg n \rceil} / 2)$		[slow growth (condition #3)]
$\leq c\, d\, f(2^{\lceil \lg n \rceil - 1})$		[simplify]
$\leq c\, d\, f(2^{\lg n})$		[use identity: $\lceil x \rceil - 1 \leq x$
		+ monotonicity (condition #2)]
$= c\, d\, f(n)$		

Let $k = c\, d$, then $T(n) \leq k\, f(n)$ forall $n > n_0$ (not just powers of 2)

Graphical idea in 4 simple steps:



Where Sloppiness Theorem does and does not apply

Sloppiness theorem DOES apply to all recurrences handled by the Master's theorem. You can pretend that you are using the Master Theorem only for n that are powers of 2, and the result will generalize to all n. This means that you can disregard floors and ceilings that are applied to n/b.

Sloppiness theorem DOES NOT apply to functions which do not "grow slowly". Though such recurrences could not be solved with the Master Theorem, they do exist. For example:

$T1(n) = n * T1(\lceil n/2 \rceil)$

$T2(n) = n * T2(\lfloor n/2 \rfloor)$

T1 is NOT $\Theta(T2(n))$

Reasoning: both T1 and T2 are exponential, and floor/ceil makes difference. Proof:

Consider n odd. Then:

$T1(n) = n * (n+1)/2 * T1(\lceil (n+1)/4 \rceil)$

$T2(n) = n * (n-1)/2 * T2(\lfloor (n-1)/4 \rfloor)$

$T1(n) = n*n + n/2 * T1(\lceil (n+1)/4 \rceil)$

$- \quad T2(n) = n*n - n/2 * T2(\lfloor (n-1)/4 \rfloor)$

$T1(n) - T2(n) = n * c(n) \quad c(n) \geq 1$ [can be shown by induction]

So there is a factor of n difference at any odd value.

Thus $T1(n) \neq \Theta(T2(n))$