

CSE 21—Mathematics for Algorithm and System Analysis

Spring, 2006

May 16, 2006

Day 13

Recurrence Relations

Instructor: Neil Rhodes

Methods for solving Recurrence Relations

Repeated substitution

- Substitute and look for patterns

Recursion tree

- Draw a tree and count values at each level

Master Method

- Works for most recurrences of the form $T_n = aT_{n/b} + f(n)$

Linear homogeneous with constant coefficients

- Works for recurrences of the form: $T_n = c_1T_{n-1} + c_2T_{n-2} + \dots + c_rT_{n-r}$

Induction

- Guess and prove by induction

Repeated Substitution

Substitute and look for patterns

- Often leads to summations. Must become good with evaluating!

Example with repeated substitution

Want good upper bound on how many multiplications to calculate a^n . (a, n integers)

- If n is even: $a^{n/2}a^{n/2}$
- If n is odd: $a \cdot a^{n-1}$

$$T_n = T_{n-1} + 1 \text{ if } n \text{ odd}$$

$$T_n = T_{n/2} + 1 \text{ if } n \text{ even}$$

$$T_1 = 0$$

$$T_0 = 0$$

Mergesort

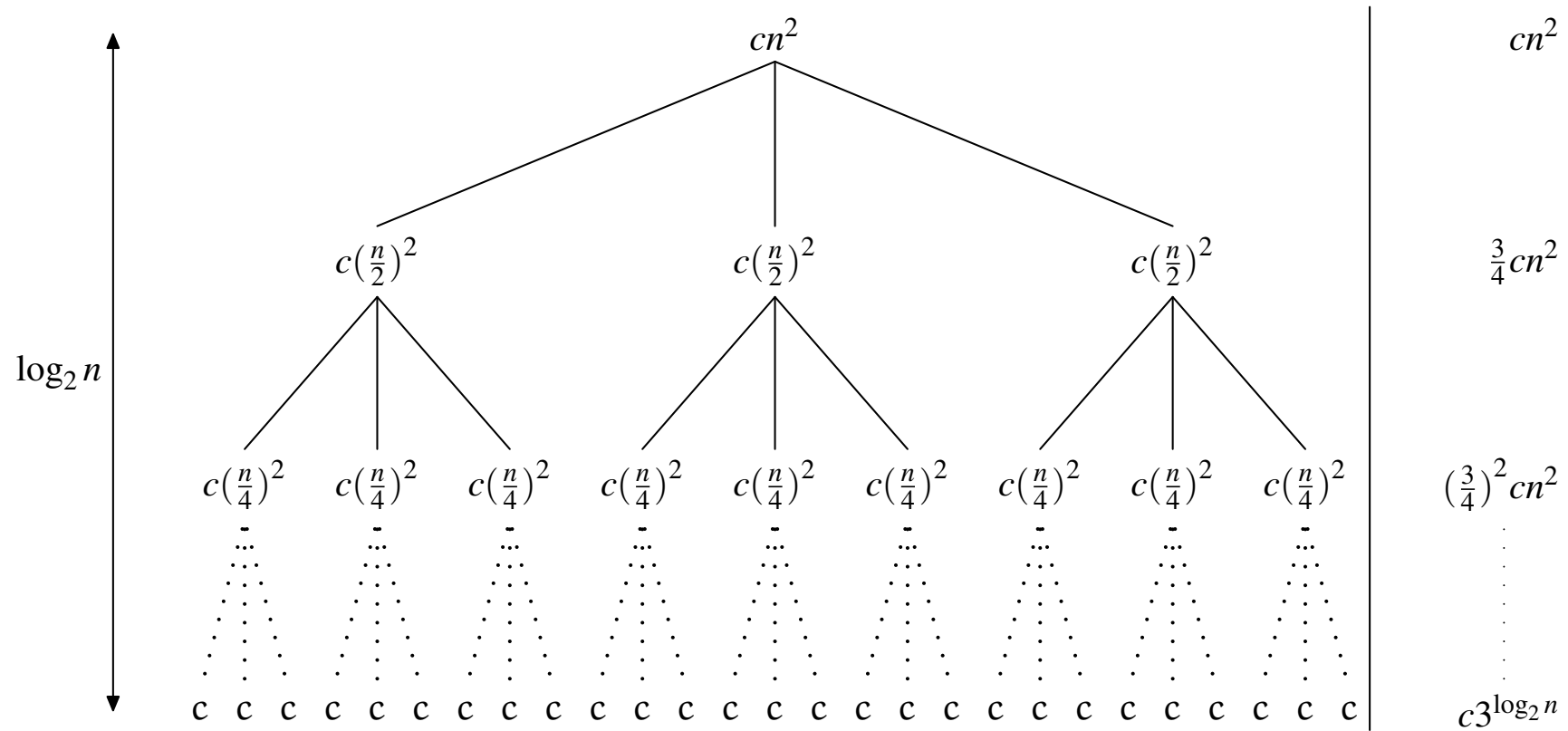
To sort a list of numbers L

- Break up list into two list of equal size: L1 and L2
- Recursively sort each list L1 and L2
- Merge together L1 and L2
- `Mergesort(L) :`
 `(L1,L2)=Split(L);`
 `return Merge(Mergesort(L1), Mergesort(L2))`

Recurrence relation?

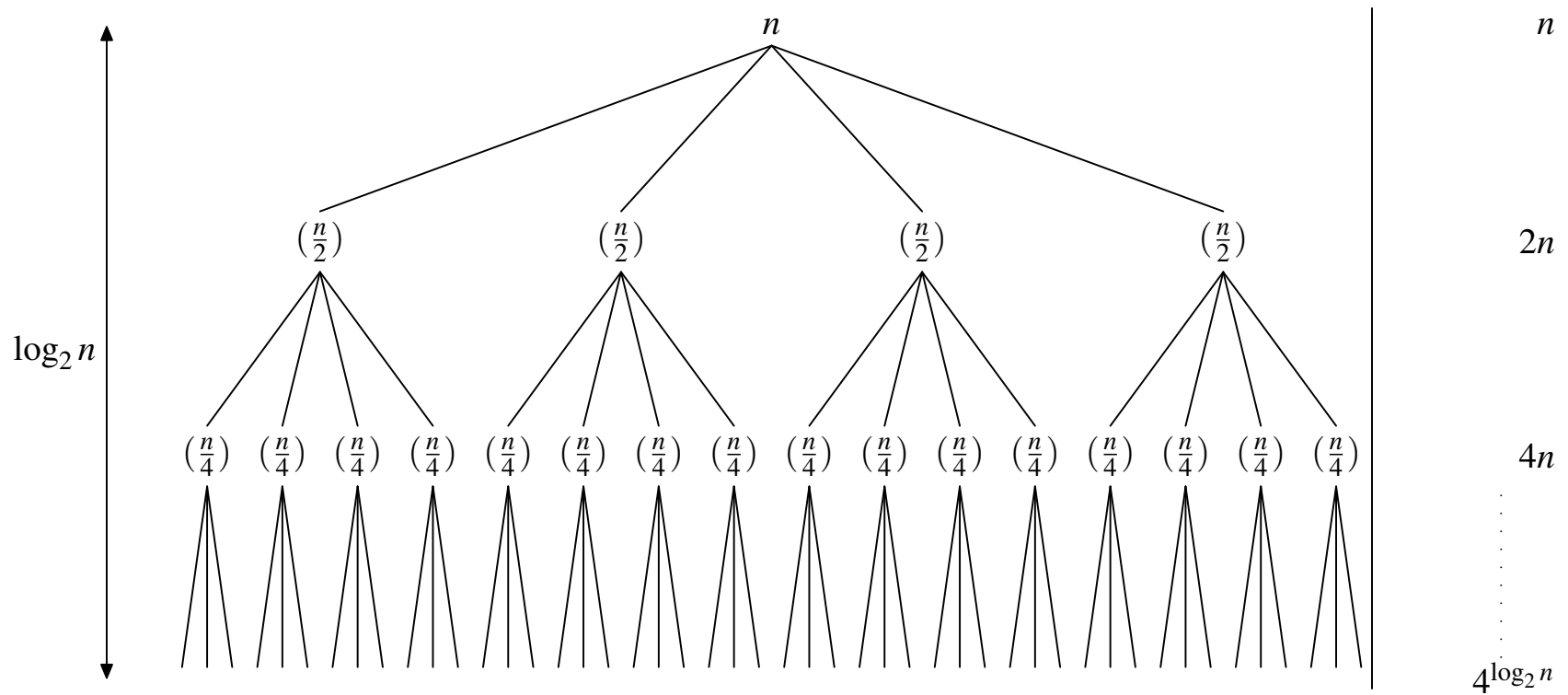
Recurrence Trees

$$T_n = 3T_{n/4} + cn^2$$



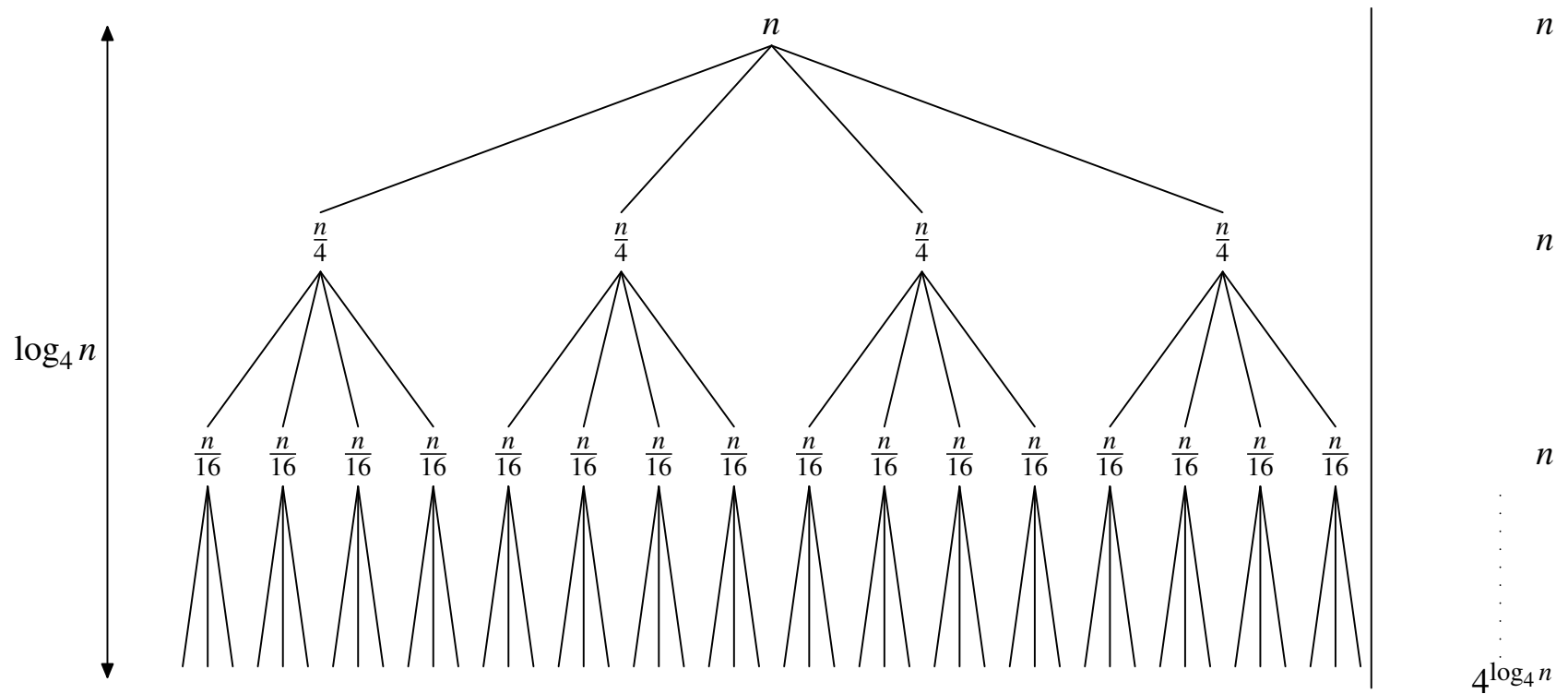
Recurrence Trees

$$T_n = 3T_{n/2} + n$$



Recurrence Trees

$$T_n = 4T_{n/4} + n$$



Master method

Given

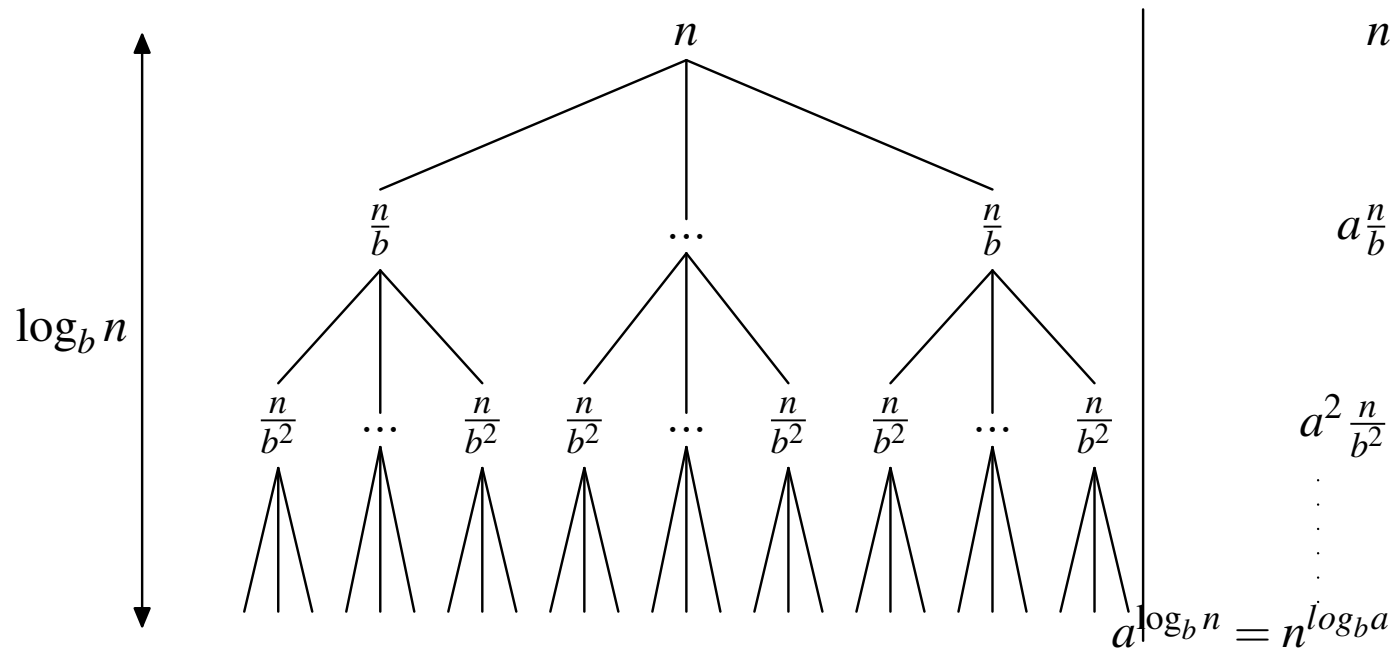
- $T_n = aT_{n/b} + f(n)$

Three general cases (in the long run):

- $f(n)$ grows faster than $n^{\log_k c}$
 - a_n grows proportional to $f(n)$
- $f(n)$ grows slower than $n^{\log_k c}$
 - a_n grows proportional to $n^{\log_k c}$
- $f(n)$ grows proportional to $n^{\log_k c}$
 - a_n grows proportional to $\log_k n \cdot n^{\log_k c}$

Limited Master Method

$$T_n = aT_{n/b} + n$$



$$\begin{aligned}
 T(n) &= n + \frac{a}{b}n + \left(\frac{a}{b}\right)^2 n + \dots + a^{\log_b n} \\
 &= \sum_{i=0}^{\log_b n} \left(\frac{a}{b}\right)^i n = n \sum_{i=0}^{\log_b n} \left(\frac{a}{b}\right)^i
 \end{aligned}$$

Limited Master Method

Evaluate : $n \sum_{i=0}^{\log_b n} \left(\frac{a}{b}\right)^i$

- Case 1: $a/b < 1$
- Case 2: $a/b > 1$
- Case 3: $a/b = 1$

Examples of Limited Master Method

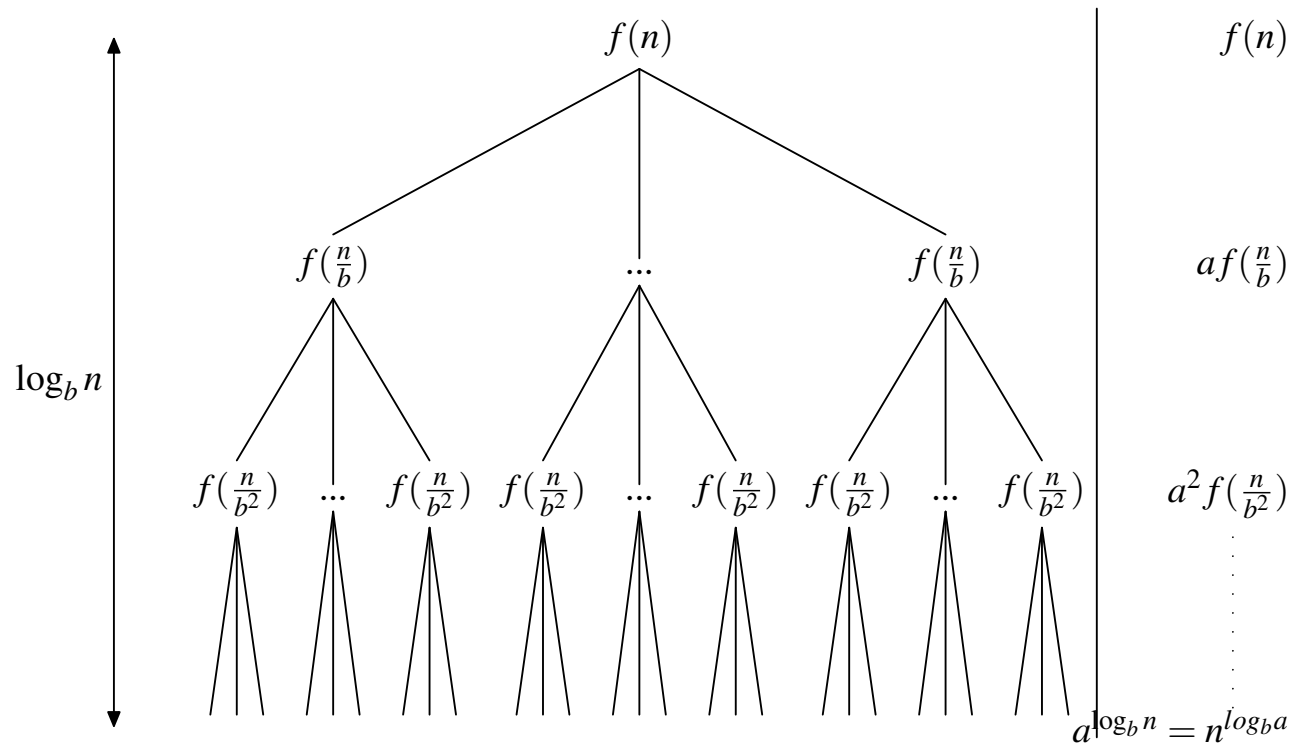
$$T_n = 3T_{n/2} + n$$

$$T_n = 4T_{n/4} + n$$

$$T_n = 4T_{n/5} + n$$

Master Method

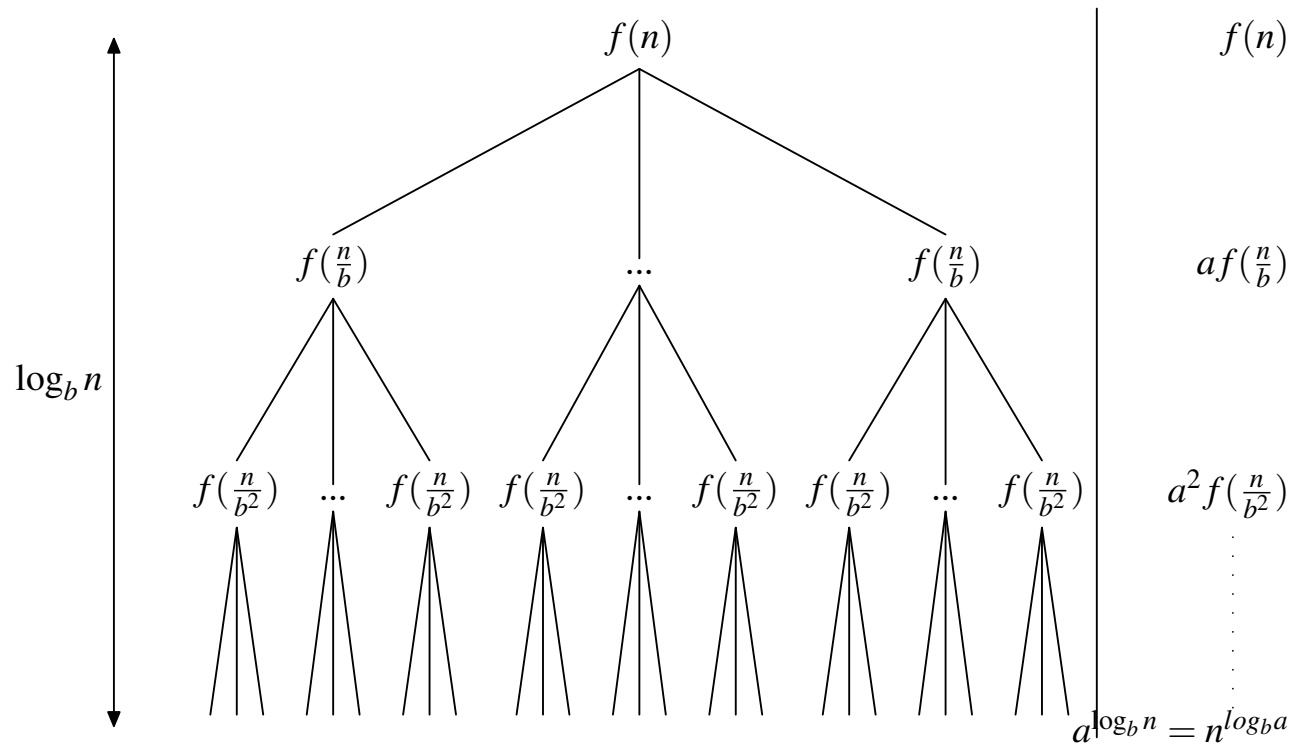
$$T_n = aT_{n/b} + f(n)$$



Case 1: $f(n)$ "larger" than $n^{\log_b a}$

Master Method

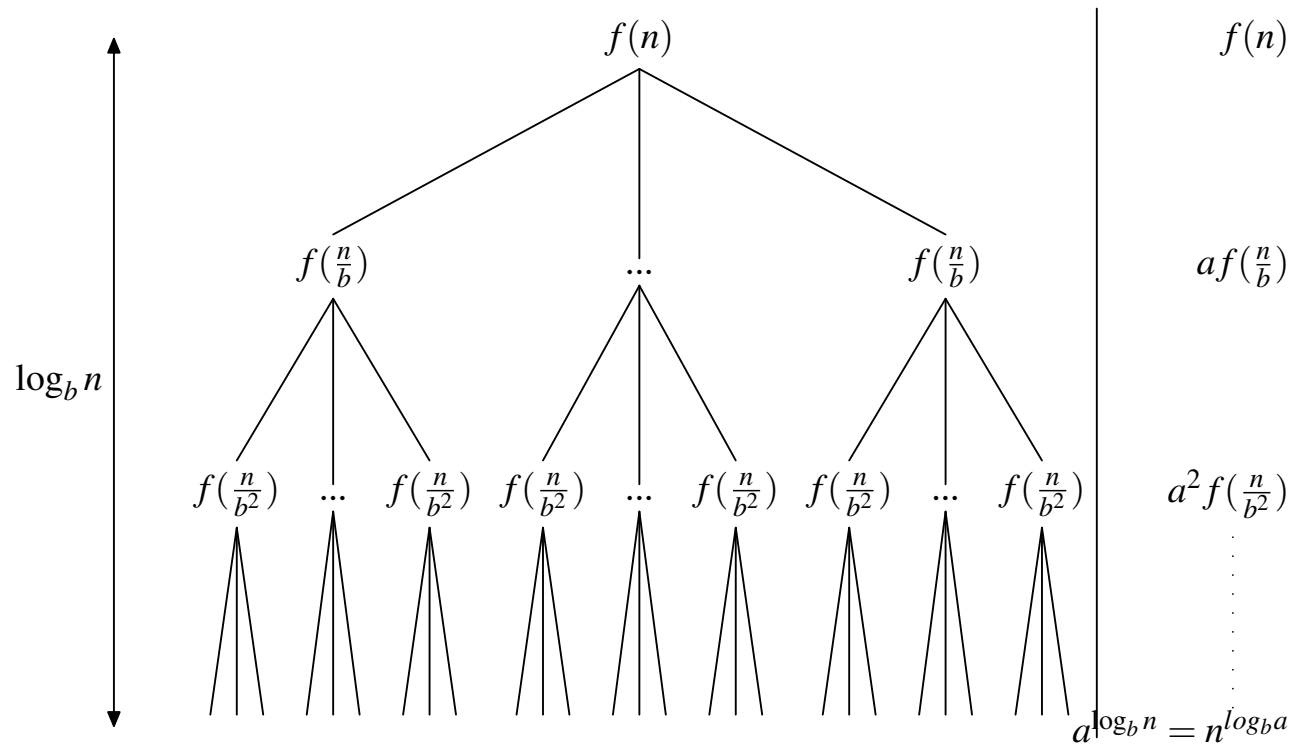
$$T_n = aT_{n/b} + f(n)$$



Case 2: $f(n)$ "smaller" than $n^{\log_b a}$

Master Method

$$T_n = aT_{n/b} + f(n)$$



Case 3: $f(n)$ "equal" $n^{\log_b a}$

Master Method Summary

Given

- $T_n = aT_{n/b} + f(n)$

Three general cases (in the long run):

- $f(n)$ grows faster than $n^{\log_b a}$
 - T_n grows proportional to $f(n)$
- $f(n)$ grows slower than $n^{\log_b a}$
 - T_n grows proportional to $n^{\log_b a}$
- $f(n)$ grows proportional to $n^{\log_b a}$
 - T_n grows proportional to $\log_b n \cdot n^{\log_b a}$

Master Method

Restrictions

- “Smaller” and “larger” mean polynomially smaller and larger
 - Must be some factor of n^ϵ different
- “Equal” means within a constant factor
- “Larger” also requires $f(n)$ is regular
 - As long as $f(n)$ is polynomial, it's regular

$f(n)$ may be in none of the three cases!

Examples of Master Method

$$T_n = 3T_{n/2} + n^2$$

$$T_n = 3T_{n/2} + n^{1.4}$$

$$T_n = 16T_{n/4} + 10n^2 + 3n + 5$$