Asymptotic Efficiency of Recurrences

- Find the asymptotic bounds of recursive equations.
  - Substitution method
    - domain transformation
    - Changing variable
  - Recursive tree method
  - Master method (master theorem)
    - Provides bounds for: $T(n) = aT(n/b)+f(n)$ where
      - $a \geq 1$ (the number of subproblems).
      - $b>1$, (n/b is the size of each subproblem).
      - $f(n)$ is a given function.

Recurrences

- MERGE-SORT
  - Contains details:
    - $T(n) = \begin{cases} \Theta(1) & \text{if } n=1 \\ T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) & \text{if } n>1 \end{cases}$
    - Ignore details, $T(n) = 2T(n/2)+ \Theta(n)$.
  - $T(n) = \begin{cases} \Theta(1) & \text{if } n=1 \\ 2T(n/2)+ \Theta(n) & \text{if } n>1 \end{cases}$

The Substitution Method

- Two steps:
  1. Guess the form of the solution.
     - By experience, and creativity.
     - By some heuristics.
       - If a recurrence is similar to one you have seen before.
         - » $T(n)=2T(\lfloor n/2 \rfloor+17)+n$, similar to $T(n)=2T(\lfloor n/2 \rfloor)+n$, , guess $O(n\lg n)$.
       - Prove loose upper and lower bounds on the recurrence and then reduce the range of uncertainty.
         - » For $T(n)=2T(\lfloor n/2 \rfloor)+n$, prove lower bound $T(n)=\Omega(n)$, and prove upper bound $T(n)=O(n^2)$, then guess the tight bound is $T(n)=O(n\lg n)$.
     - By recursion tree.
  2. Use mathematical induction to find the constants and show that the solution works.

Solve $T(n)=2T(\lfloor n/2 \rfloor)+n$

- Guess the solution: $T(n)=O(n\lg n)$,
  - i.e., $T(n)\leq cn\lg n$ for some c.
- Prove the solution by induction:
  - Suppose this bound holds for $\lfloor n/2 \rfloor$, i.e.,
    - $T(\lfloor n/2 \rfloor)\leq c\lfloor n/2 \rfloor \lg (\lfloor n/2 \rfloor)$.
  - $T(n) \leq 2(c\lfloor n/2 \rfloor \lg (\lfloor n/2 \rfloor))+n$
    - $\leq cn \lg (n/2))+n$
    - $= cn \lg n - cn \lg 2 +n$
    - $= cn \lg n - cn +n$
    - $\leq cn \lg n$ (as long as $c\geq 1$)

Question: Is the above proof complete? Why?

Boundary (base) Condition

- In fact, $T(n) = 1$ if $n=1$, i.e., $T(1)=1$.
- However, $cn\lg n = c\times 1\times \lg 1 = 0$, which is odd with $T(1)=1$.
- Take advantage of asymptotic notation: it is required $T(n)\leq cn\lg n$ hold for $n\geq n_0$ where n_0 is a constant of our choosing.
- Select $n_0 = 2$, thus, $n=2$ and $n=3$ as our induction bases. It turns out any $c \geq 2$ suffices for base cases of $n=2$ and $n=3$ to hold.

Subtleties

- Guess is correct, but induction proof not work.
- Problem is that inductive assumption not strong enough.
- Solution: revise the guess by subtracting a lower-order term.
- Example: $T(n)=T(\lfloor n/2\rfloor)+T(\lceil n/2\rceil)+1$.
  - Guess $T(n)=O(n)$, i.e., $T(n) \leq cn$ for some c.
  - However, $T(n) \leq c\lfloor n/2\rfloor + c\lceil n/2\rceil + 1 = cn+1$, which does not imply $T(n) \leq cn$ for any c.
  - Attempting $T(n)=O(n^2)$ will work, but overkill.
  - New guess $T(n) \leq cn - b$ will work as long as $b \geq 1$.
  - (Prove it in an exact way).

Avoiding Pitfall

- It is easy to guess $T(n)=O(n)$ (i.e., $T(n) \le cn$) for $T(n)=2T(\lfloor n/2 \rfloor)+n$.
- And wrongly prove:
  - $T(n) \le 2(c\lfloor n/2 \rfloor)+n$
    - $\le cn+n$
    - $=O(n)$. ← wrongly !!!!
- Problem is that it does not prove the *exact form* of $T(n) \le cn$.

Find bound, ceiling, floor, lower term– domain transformation

- Find the bound: $T(n)=2T(n/2)+n$ ($O(n\log n)$)
- How about $T(n)=2T(\lfloor n/2 \rfloor)+n$?
- How about $T(n)=2T(\lceil n/2 \rceil)+n$?
  - $T(n) \le 2T(n/2+1)+n$
  - Domain transformation
    - Set $S(n)=T(n+a)$ and assume $S(n) \le 2S(n/2)+O(n)$ (so $S(n)=O(n\log n)$)
    - $S(n) \le 2S(n/2)+O(n) \rightarrow T(n+a) \le 2T(n/2+a)+O(n)$
    - $T(n) \le 2T(n/2+1)+n \rightarrow T(n+a) \le 2T((n+a)/2+1)+n+a$
    - Thus, set $n/2+a=(n+a)/2+1$, get $a=2$.
    - so $T(n)=S(n-2)=O((n-2)\log(n-2)) = O(n\log n)$.
- How about $T(n)=2T(n/2+19)+n$?
  - Set $S(n)=T(n+a)$ and get a=38.
- As a result, ceiling, floor, and lower terms will not affect.
  - Moreover, the master theorem also provides proof for this.

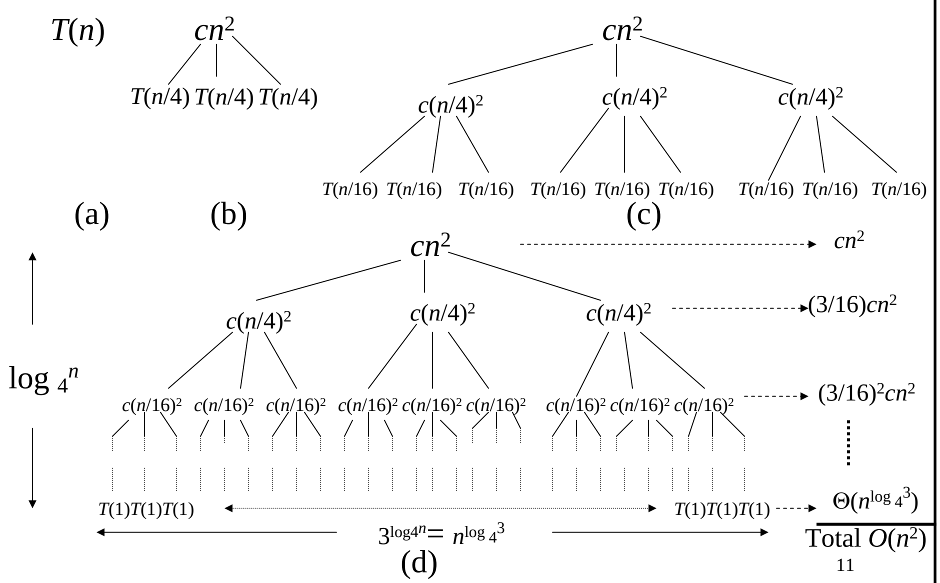
Changing Variables

- Suppose $T(n)=2T(\sqrt{n})+\lg n$.
- Rename $m=\lg n$. so $T(2^m)=2T(2^{m/2})+m$.
- Domain transformation:
  - $S(m)=T(2^m)$, so $S(m)=2S(m/2)+m$.
  - Which is similar to $T(n)=2T(n/2)+n$.
  - So the solution is $S(m)=O(m \lg m)$.
  - Changing back to $T(n)$ from $S(m)$, the solution is $T(n) =T(2^m)=S(m)=O(m \lg m)=O(\lg n \lg \lg n)$.

The Recursion-tree Method

- Idea:
  - Each node represents the cost of a single subproblem.
  - Sum up the costs with each level to get level cost.
  - Sum up all the level costs to get total cost.
- Particularly suitable for divide-and-conquer recurrence.
- Best used to generate a good guess, tolerating "sloppiness".
- If trying carefully to draw the recursion-tree and compute cost, then used as direct proof.

Recursion Tree for $T(n)=3T(\lfloor n/4 \rfloor)+\Theta(n^2)$



Solution to $T(n)=3T(\lfloor n/4 \rfloor)+\Theta(n^2)$

- The height is $\log_4 n$,
- #leaf nodes = $3^{\log_4 n} = n^{\log_4 3}$. Leaf node cost: $T(1)$.
- Total cost $T(n)=cn^2+(3/16)\,cn^2+(3/16)^2\,cn^2+$
  $\cdots+(3/16)^{\log_4 n-1}\,cn^2+\Theta(n^{\log_4 3})$
  $=(1+3/16+(3/16)^2+\cdots+(3/16)^{\log_4 n-1})\,cn^2+\Theta(n^{\log_4 3})$
  $<(1+3/16+(3/16)^2+\cdots+(3/16)^m+\cdots)\,cn^2+\Theta(n^{\log_4 3})$
  $=(1/(1-3/16))\,cn^2+\Theta(n^{\log_4 3})$
  $=16/13cn^2+\Theta(n^{\log_4 3})$
  $=O(n^2)$.

Prove the above Guess

- $T(n)=3T(\lfloor n/4 \rfloor)+\Theta(n^2) = O(n^2)$.
- Show $T(n) \le dn^2$ for some d.
- $T(n) \le 3(d\,(\lfloor n/4 \rfloor)^2) + cn^2$

  $\le 3(d\,(n/4)^2) + cn^2$

  $= 3/16(dn^2) + cn^2$

  $\le dn^2$, as long as $d \ge (16/13)c$.

One more example

- $T(n)=T(n/3)+T(2n/3)+O(n)$.
- Construct its recursive tree (Figure 4.2, page 71).
- $T(n)=O(cn\lg_{3/2}{}^{n}) = O(n\lg n)$.
- Prove $T(n) \le dn\lg n$.

Recursion Tree of $T(n)=T(n/3)+T(2n/3)+O(n)$

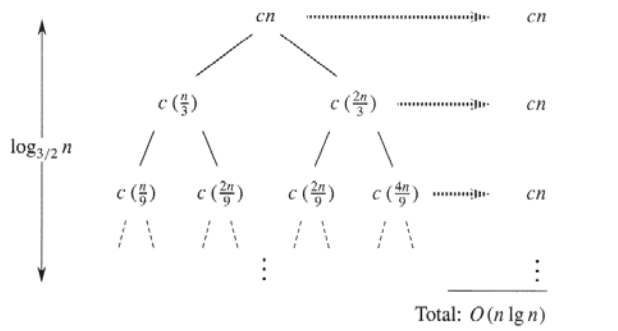


Figure 4.2 A recursion tree for the recurrence $T(n) = T(n/3) + T(2n/3) + cn$.

Master Method/Theorem

- Theorem 4.1 (page 73)
  - for $T(n) = aT(n/b)+f(n)$, n/b may be $\lceil n/b \rceil$ or $\lfloor n/b \rfloor$.
  - where $a \geq 1$, $b>1$ are positive integers, $f(n)$ be a non-negative function.
  1. If $f(n)=O(n^{\log_b a-\varepsilon})$ for some $\varepsilon>0$, then $T(n)=\Theta(n^{\log_b a})$.
  2. If $f(n)=\Theta(n^{\log_b a})$, then $T(n)=\Theta(n^{\log_b a} \lg n)$.
  3. If $f(n)=\Omega(n^{\log_b a+\varepsilon})$ for some $\varepsilon>0$, and if $af(n/b) \leq cf(n)$ for some $c<1$ and all sufficiently large n, then $T(n)=\Theta(f(n))$.

Implications of Master Theorem

- Comparison between $f(n)$ and $n^{\log_b a}$ (<,=,>)
- Must be asymptotically smaller (or larger) by a polynomial, i.e., n^{ε} for some $\varepsilon>0$.
- In case 3, the "regularity" must be satisfied, i.e., $af(n/b) \leq cf(n)$ for some $c<1$.
- There are gaps
  - between 1 and 2: $f(n)$ is smaller than $n^{\log_b a}$, but not polynomially smaller.
  - between 2 and 3: $f(n)$ is larger than $n^{\log_b a}$, but not polynomially larger.
  - in case 3, if the "regularity" fails to hold.

Application of Master Theorem

- $T(n) = 9T(n/3)+n$;
  - $a=9, b=3, f(n) = n$
  - $n^{\log_b a} = n^{\log_3 9} = \Theta(n^2)$
  - $f(n)=O(n^{\log_3 9-\varepsilon})$ for $\varepsilon=1$
  - By case 1, $T(n) = \Theta(n^2)$.
- $T(n) = T(2n/3)+1$
  - $a=1, b=3/2, f(n) = 1$
  - $n^{\log_b a} = n^{\log_{3/2} 1} = \Theta(n^0) = \Theta(1)$
  - By case 2, $T(n)= \Theta(\lg n)$.

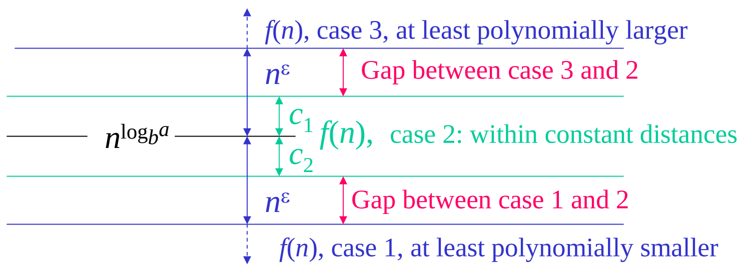
Application of Master Theorem

- $T(n) = 3T(n/4)+n\lg n$;
  - a=3,b=4, $f(n) = n\lg n$
  - $n^{\log_b a} = n^{\log_4 3} = \Theta(n^{0.793})$
  - $f(n)= \Omega(n^{\log_4 3+\varepsilon})$ for $\varepsilon\approx 0.2$
  - Moreover, for large n, the "regularity" holds for c=3/4.
    - $af(n/b) = 3(n/4)\lg(n/4) \le (3/4)n\lg n = cf(n)$
  - By case 3, $T(n) = \Theta(f(n)) = \Theta(n\lg n)$.

Exception to Master Theorem

- $T(n) = 2T(n/2)+n\lg n$;
  - a=2,b=2, $f(n) = n\lg n$
  - $n^{\log_b a} = n^{\log_2 2} = \Theta(n)$
  - $f(n)$ is asymptotically larger than $n^{\log_b a}$, but not polynomially larger because
  - $f(n)/n^{\log_b a} = \lg n$, which is asymptotically less than n^{ε} for any $\varepsilon>0$.
  - Therefore,this is a gap between 2 and 3.

Where Are the Gaps



Note: 1. for case 3, the regularity also must hold.
2. if $f(n)$ is $\lg n$ smaller, then fall in gap in 1 and 2
3. if $f(n)$ is $\lg n$ larger, then fall in gap in 3 and 2
4. if $f(n)=\Theta(n^{\log_b a}\lg^k n)$, then $T(n)=\Theta(n^{\log_b a}\lg^{k+1} n)$. (as exercise)

Proof of Master Theorem

- The proof for the exact powers, $n=b^k$ for $k\geq 1$.
- Lemma 4.2
  - for $T(n) = \begin{cases} \Theta(1) & \text{if } n=1 \\ aT(n/b)+f(n) & \text{if } n=b^k \text{ for } k\geq 1 \end{cases}$
  - where $a \geq 1$, $b>1$, $f(n)$ be a nonnegative function,
  - Then
  - $T(n) = \Theta(n^{\log_b a})+\sum_{j=0}^{\log_b n-1} a^j f(n/b^j)$
- Proof:
  - By iterating the recurrence
  - By recursion tree (See figure 4.3)

Recursion tree for $T(n)=aT(n/b)+f(n)$

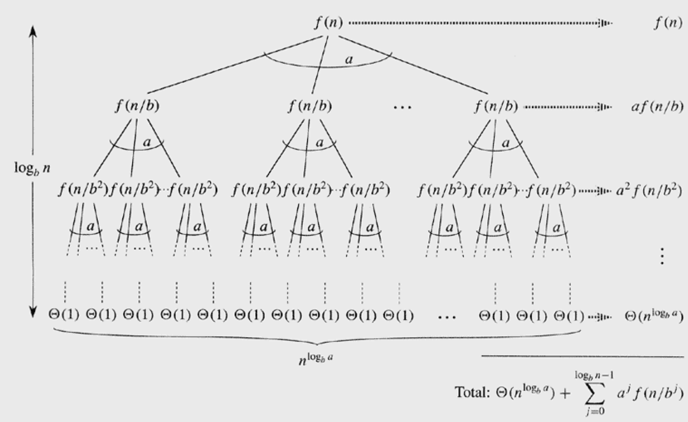


Figure 4.3 The recursion tree generated by $T(n) = aT(n/b) + f(n)$. The tree is a complete a-ary tree with $n^{\log_b a}$ leaves and height $\log_b n$. The cost of each level is shown at the right, and their sum is given in equation (4.6).

Proof of Master Theorem (cont.)

- Lemma 4.3:
  - Let $a \geq 1$, $b>1$, $f(n)$ be a nonnegative function defined on exact power of b, then
  - $g(n)= \sum_{j=0}^{\log_b n-1} a^j f(n/b^j)$ can be bounded for exact power of b as:

1. If $f(n)=O(n^{\log_b a-\varepsilon})$ for some $\varepsilon>0$, then $g(n)= O(n^{\log_b a})$.
2. If $f(n)= \Theta(n^{\log_b a})$, then $g(n)= \Theta(n^{\log_b a} \lg n)$.
3. If $f(n)= \Omega(n^{\log_b a+\varepsilon})$ for some $\varepsilon>0$ and if $af(n/b) \leq cf(n)$ for some $c<1$ and all sufficiently large $n \geq b$, then $g(n)= \Theta(f(n))$.

Proof of Lemma 4.3

- For case 1: $f(n)=O(n^{\log_b a-\varepsilon})$ implies $f(n/b^j)=O((n/b^j)^{\log_b a-\varepsilon})$, so
- $g(n)=\sum_{j=0}^{\log_b n-1} a^j f(n/b^j) = O(\sum_{j=0}^{\log_b n-1} a^j (n/b^j)^{\log_b a-\varepsilon})$
- $= O(n^{\log_b a-\varepsilon} \sum_{j=0}^{\log_b n-1} a^j/(b^{\log_b a-\varepsilon})^j) = O(n^{\log_b a-\varepsilon} \sum_{j=0}^{\log_b n-1} a^j/(a^j(b^{-\varepsilon})^j))$
- $= O(n^{\log_b a-\varepsilon} \sum_{j=0}^{\log_b n-1} (b^{\varepsilon})^j) = O(n^{\log_b a-\varepsilon} (((b^{\varepsilon})^{\log_b n}-1)/(b^{\varepsilon}-1))$
- $= O(n^{\log_b a-\varepsilon} (((b^{\log_b n})^{\varepsilon}-1)/(b^{\varepsilon}-1)))=O(n^{\log_b a} n^{-\varepsilon} (n^{\varepsilon}-1)/(b^{\varepsilon}-1))$
- $= O(n^{\log_b a})$

Proof of Lemma 4.3(cont.)

- For case 2: $f(n)=\Theta(n^{\log_b a})$ implies $f(n/b^j)=\Theta((n/b^j)^{\log_b a})$, so
- $g(n)=\sum_{j=0}^{\log_b n-1} a^j f(n/b^j) = \Theta(\sum_{j=0}^{\log_b n-1} a^j (n/b^j)^{\log_b a})$
- $= \Theta(n^{\log_b a} \sum_{j=0}^{\log_b n-1} a^j/(b^{\log_b a})^j) = \Theta(n^{\log_b a} \sum_{j=0}^{\log_b n-1} 1)$
- $= \Theta(n^{\log_b a} \log_b n) = \Theta(n^{\log_b a} \lg n)$

Proof of Lemma 4.3(cont.)

- For case 3:
  - Since $g(n)$ contains $f(n)$, $g(n) = \Omega(f(n))$
  - Since $af(n/b) \le cf(n)$, $a^j f(n/b^j) \le c^j f(n)$, why???
  - $g(n)= \sum_{j=0}^{\log_b n-1} a^j f(n/b^j) \le \sum_{j=0}^{\log_b n-1} c^j f(n) \le f(n) \sum_{j=0}^{\infty} c^j$
  - $=f(n)(1/(1-c)) =O(f(n))$
  - Thus, $g(n)=\Theta(f(n))$

Proof of Master Theorem (cont.)

- Lemma 4.4:
  - for $T(n) = \begin{cases} \Theta(1) & \text{if } n=1 \\ aT(n/b)+f(n) & \text{if } n=b^k \text{ for } k\ge 1 \end{cases}$
  - where $a \ge 1$, $b>1$, $f(n)$ be a nonnegative function,

1. If $f(n)=O(n^{\log_b a-\varepsilon})$ for some $\varepsilon>0$, then $T(n)= \Theta(n^{\log_b a})$.
2. If $f(n)= \Theta(n^{\log_b a})$, then $T(n)= \Theta(n^{\log_b a} \lg n)$.
3. If $f(n)=\Omega(n^{\log_b a+\varepsilon})$ for some $\varepsilon>0$, and if $af(n/b) \le cf(n)$ for some $c<1$ and all sufficiently large n, then $T(n)= \Theta(f(n))$.

Proof of Lemma 4.4 (cont.)

- Combine Lemma 4.2 and 4.3,
  - For case 1:
    - $T(n)=\Theta(n^{\log_b a})+O(n^{\log_b a})=\Theta(n^{\log_b a})$.
  - For case 2:
    - $T(n)=\Theta(n^{\log_b a})+\Theta(n^{\log_b a}\lg n)=\Theta(n^{\log_b a}\lg n)$.
  - For case 3:
    - $T(n)=\Theta(n^{\log_b a})+\Theta(f(n))=\Theta(f(n))$ because $f(n)=\Omega(n^{\log_b a+\varepsilon})$.

Floors and Ceilings

- $T(n)=aT(\lfloor n/b\rfloor)+f(n)$ and $T(n)=aT(\lceil n/b\rceil)+f(n)$
- Want to prove both equal to $T(n)=aT(n/b)+f(n)$
- Two results:
  - Master theorem applied to all integers n.
  - Floors and ceilings do not change the result.
    - (Note: we proved this by domain transformation too).
- Since $\lfloor n/b\rfloor\leq n/b$, and $\lceil n/b\rceil\geq n/b$, upper bound for floors and lower bound for ceiling is held.
- So prove upper bound for ceilings (similar for lower bound for floors).

Upper bound of proof for $T(n)=aT(\lceil n/b \rceil)+f(n)$

- consider sequence $n, \lceil n/b \rceil, \lceil \lceil n/b \rceil /b \rceil, \lceil \lceil \lceil n/b \rceil /b \rceil /b \rceil$, …
- Let us define n_j as follows:
- $n_j = n$ if $j=0$
- $= \lceil n_{j-1}/b \rceil$ if $j>0$
- The sequence will be $n_0, n_1, \ldots, n_{\lfloor \log_b n \rfloor}$
- Draw recursion tree:

Recursion tree of $T(n)=aT(\lceil n/b \rceil)+f(n)$

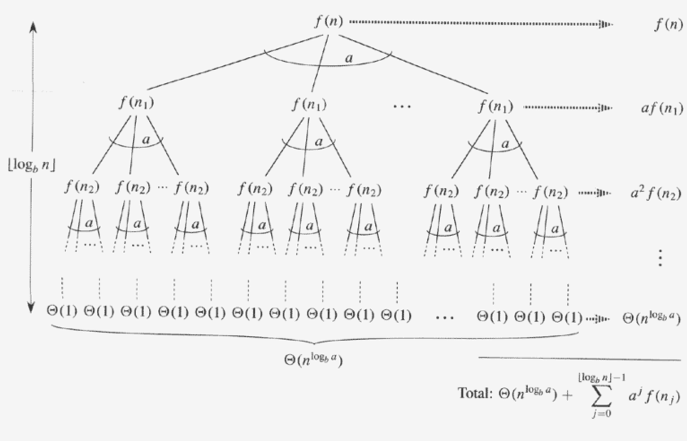


Figure 4.4 The recursion tree generated by $T(n) = aT(\lceil n/b \rceil) + f(n)$. The recursive argument n_j is given by equation (4.12).

The proof of upper bound for ceiling

– $T(n) = \Theta(n^{\log_b a}) + \sum_{j=0}^{\lfloor \log_b n \rfloor - 1} a^j f(n_j)$

– Thus similar to Lemma 4.3 and 4.4, the upper bound is proven.

The simple format of master theorem

- $T(n)=aT(n/b)+cn^k$, with a, b, c, k are positive constants, and $a\geq1$ and $b\geq2$,
- $T(n) = \begin{cases} O(n^{\log_b a}), & \text{if } a>b^k. \\ O(n^k \log n), & \text{if } a=b^k. \\ O(n^k), & \text{if } a<b^k. \end{cases}$

Summary

Recurrences and their bounds

- Substitution
- Recursion tree
- Master theorem.
- Proof of subtleties
- Recurrences that Master theorem does not apply to.