Design and Analysis of Algorithms

CSE 5311

Lecture 4 Master Theorem

Junzhou Huang, Ph.D.

Department of Computer Science and Engineering

Reviewing: Solving Recurrences

- **Recurrence**
  - The analysis of integer multiplication from last lecture required us to solve a recurrence
  - Recurrences are a major tool for analysis of algorithms
  - Divide and Conquer algorithms which are analyzable by recurrences.

- **Three steps at each level of the recursion:**
  - **Divide** the problem into a number of subproblems that are smaller instances of the same problem.
  - **Conquer** the subproblems by solving them recursively. If the subproblem sizes are small enough, however, just solve the subproblems in a straightforward manner.
  - **Combine** the solutions to the subproblems into the solution for the original problem.

Recall: Integer Multiplication

- Let X = $\boxed{A \mid B}$ and Y = $\boxed{C \mid D}$ where A,B,C and D are n/2 bit integers
- Simple Method: $XY = (2^{n/2}A+B)(2^{n/2}C+D)$
- Running Time Recurrence

$$T(n) < 4T(n/2) + \Theta(n)$$

How do we solve it?

Reviewing: Substitution Method

The most general method:

1. ***Guess*** the form of the solution.
2. ***Verify*** by induction.
3. ***Solve*** for constants.

Example: $T(n) = 4T(n/2) + \Theta(n)$

- [Assume that $T(1) = \Theta(1)$.]
- Guess $O(n^3)$. (Prove O and Ω separately.)
- Assume that $T(k) \leq ck^3$ for $k < n$.
- Prove $T(n) \leq cn^3$ by induction.

The Master Method

The master method applies to recurrences of the form

$$T(n) = a\,T(n/b) + f(n)\,,$$

where $a \geq 1$, $b > 1$, and f is asymptotically positive.

1. $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$. Then, $T(n) = \Theta(n^{\log_b a})$

2. $f(n) = \Theta(n^{\log_b a})$ for $k \geq 0$. Then, $T(n) = \Theta(n^{\log_b a} \lg n)$.

3. $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$ ***and*** $f(n)$ satisfies the ***regularity condition*** that $a\,f(n/b) \leq c\,f(n)$ for some constant $c < 1$. Then, $T(n) = \Theta(f(n))$

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, $T(n)= \Theta(\lg n)$.

6

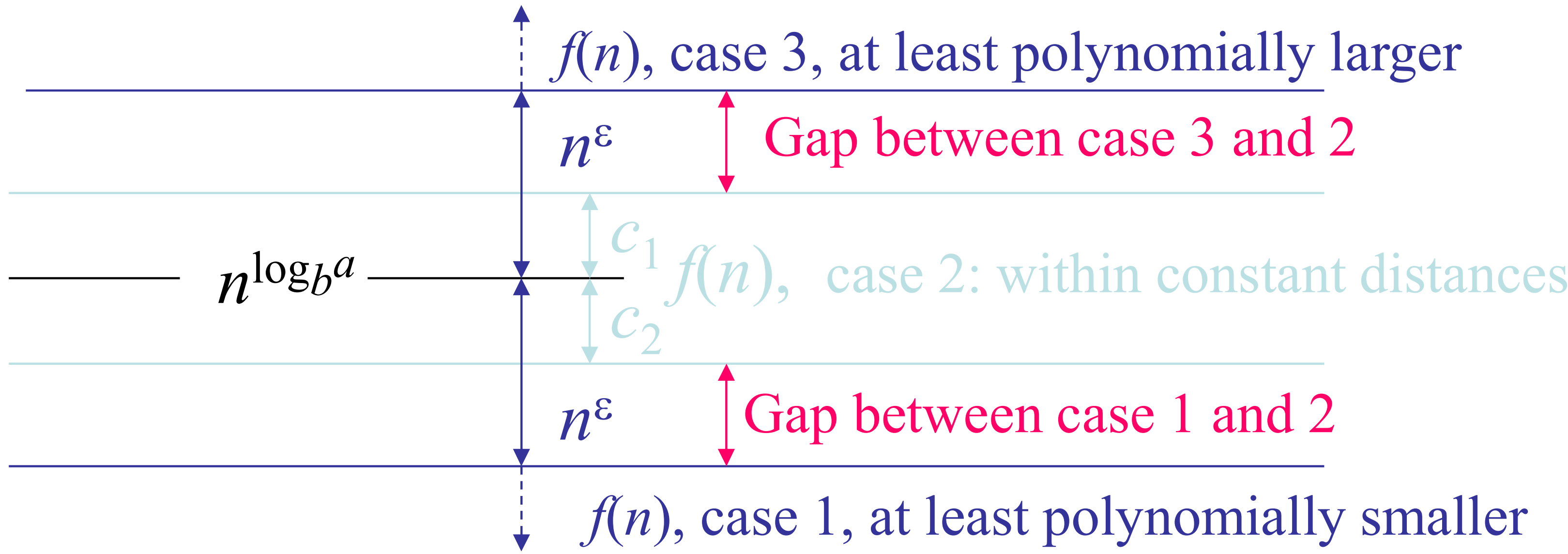
Application of Master Theorem

- $T(n) = 3T(n/4)+n\lg n$;
  - 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.
    - $af(n/b) = 3(n/4)\lg(n/4) \leq (3/4)n\lg n = cf(n)$
  - 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)

9

Master Theorem

The master method applies to recurrences of the form

$$T(n) = a\, T(n/b) + f(n)\ ,$$

where constants $a \geq 1$, $b > 1$, and f is asymptotically positive function

1. $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
2. $f(n) = O(n^{\log_b a})$ for some constant $\varepsilon > 0$, then $T(n) = \Theta(n^{\log_b a} \lg n)$
3. $f(n) = O(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$, and if $a f(n/b) \leq c f(n)$ for some constant $c < 1$, then $T(n) = \Theta(f(n))$.

How to theoretically prove it?

Proof for Exact Powers

- Suppose $n=b^k$ 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 defined on exact powers of b, 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 $T(n)=aT(n/b)+f(n)$

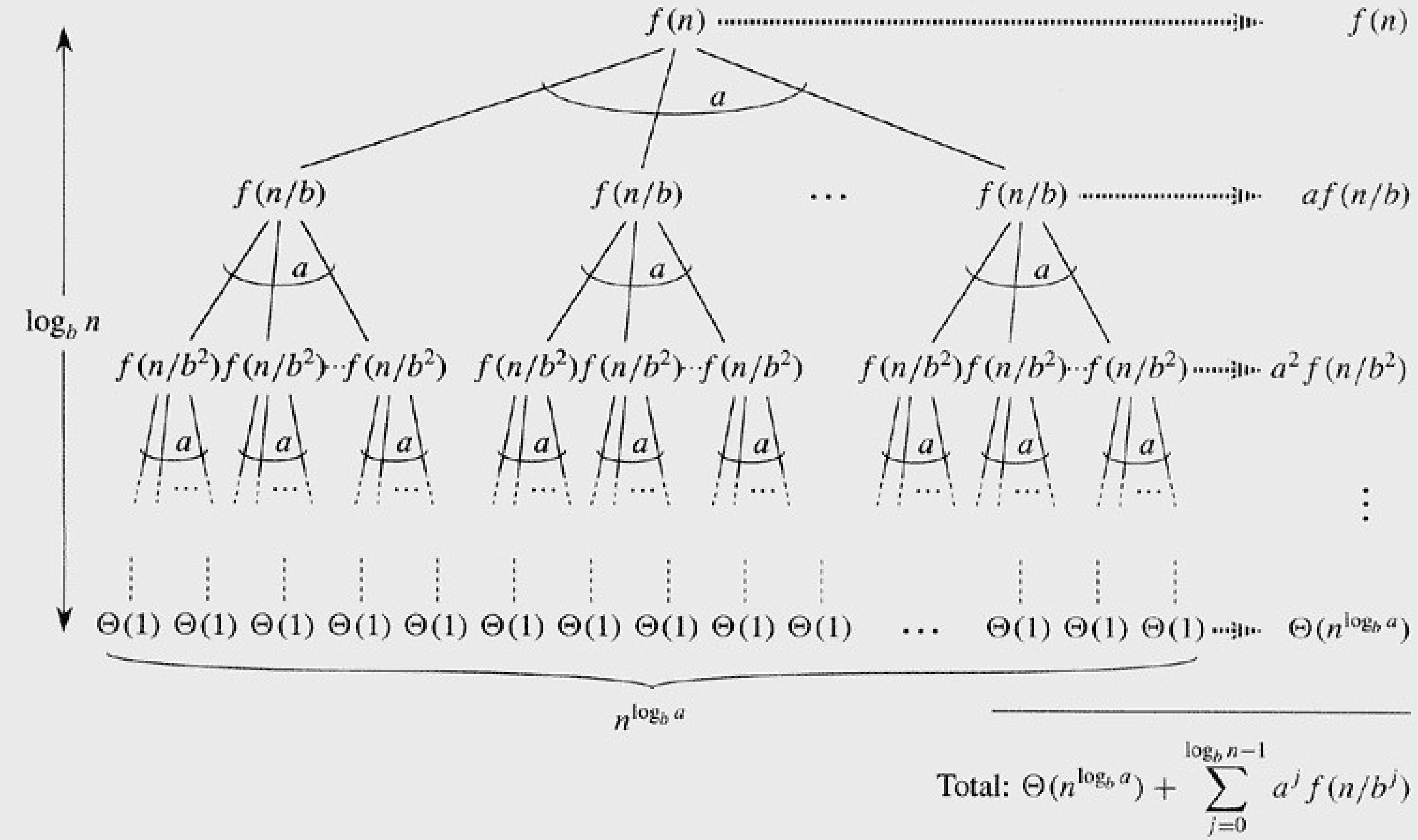


Figure 4.3 The recursion tree generated by $T(n) = aT(n/b) + f(n)$. The tree is a complete a-ary tree with $n^{\log_b a}$ leaves and height $\log_b n$. The cost of each level is shown at the right, and their sum is given in equation (4.6).

Proof for Exact Powers (cont.)

- Lemma 4.3:
  - Let constants $a \geq 1$, $b>1$, $f(n)$ 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 asymptotically for exact power of b as follows:

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
- $g(n)= \sum_{j=0}^{\log_b n-1} a^j f(n/b^j) = O\left(\sum_{j=0}^{\log_b n-1} a^j (n/b^j)^{\log_b a-\varepsilon}\right)$
- $= O\left(n^{\log_b a-\varepsilon} \sum_{j=0}^{\log_b n-1} a^j/(b^{\log_b a-\varepsilon})^j\right) = O\left(n^{\log_b a-\varepsilon} \sum_{j=0}^{\log_b n-1} a^j/(a^j(b^{-\varepsilon})^j)\right)$
- $= O\left(n^{\log_b a-\varepsilon} \sum_{j=0}^{\log_b n-1} (b^{\varepsilon})^j\right) = O\left(n^{\log_b a-\varepsilon} (((b^{\varepsilon})^{\log_b n}-1)/(b^{\varepsilon}-1))\right)$
- $= O(n^{\log_b a-\varepsilon} (((b^{\log_b n})^{\varepsilon}-1)/(b^{\varepsilon}-1)))$
- $= O(n^{\log_b a} n^{-\varepsilon} (n^{\varepsilon}-1)/(b^{\varepsilon}-1))$
- $= O(n^{\log_b a})$

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

- $g(n) = \sum_{j=0}^{\log_b n - 1} a^j f(n/b^j) = \Theta\left(\sum_{j=0}^{\log_b n - 1} a^j (n/b^j)^{\log_b a}\right)$

- $= \Theta\left(n^{\log_b a} \sum_{j=0}^{\log_b n - 1} a^j / (b^{\log_b a})^j\right) = \Theta\left(n^{\log_b a} \sum_{j=0}^{\log_b n - 1} 1\right)$

- $= \Theta(n^{\log_b a} \log_b n) = \Theta(n^{\log_b a} \lg n)$

Proof of Lemma 4.3(cont.)

- For case 3:
  - Since $g(n)$ contains $f(n)$, $g(n) = \Omega(f(n))$
  - Since $a\, f(n/b) \le c\, f(n)$, so $f(n/b) \le (c/a)\, f(n)$,
  - *Iterating j times,* $f(n/b^j) \le (c/a)^j f(n)$, thus $a^j f(n/b^j) \le c^j f(n)$
  - $$g(n) = \sum_{j=0}^{\log_b n-1} a^j f(n/b^j) \le \sum_{j=0}^{\log_b n-1} c^j f(n) \le f(n) \sum_{j=0}^{\infty} c^j = f(n)\,(1/(1-c))$$

    $$= O(f(n))$$
  - Thus, $g(n)=\Theta(f(n))$

Proof for Exact Powers (cont.)

- Lemma 4.4:
  - for $T(n) = \begin{cases} \Theta(1) & \text{if } n=1 \\ aT(n/b)+f(n) & \text{if } n=b^k \text{ for } k\geq 1 \end{cases}$
  - where $a \geq 1$, $b>1$, $f(n)$ 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 $af(n/b)\leq cf(n)$ 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 ($n \neq b^k$ for $k \geq 1$)

- $T(n)=a\, T(\lfloor n/b \rfloor)+f(n)$ and $T(n)=a\, T(\lceil n/b \rceil)+f(n)$
- Want to prove both equal to $T(n)=a\, T(n/b)+f(n)$
- 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 $n, \lceil n/b \rceil, \lceil \lceil n/b \rceil / b \rceil, \lceil \lceil \lceil n/b \rceil / b \rceil / b \rceil, \ldots$
- Let us define n_j as follows:
- $n_j = n \qquad \text{if } j = 0$
- $\qquad = \lceil n_{j-1}/b \rceil \quad \text{if } j > 0$
- The sequence will be $n_0, n_1, \ldots, n_{\lfloor \log_b n \rfloor}$

$$n_0 <= n$$
$$n_1 <= n/b+1$$
$$n_2 <= n/b^2+n/b+1$$
$$\ldots$$
$$n_j <= n/b^j + \sum_{i=0}^{j-1} 1/b^i$$
$$< \quad n/b^j + b/(b-1)$$

Let $j = \lfloor \log_b n \rfloor$, **then**

$$n_{\lfloor \log_b n \rfloor} < n / b^{\lfloor \log_b n \rfloor} + b/(b-1)$$

$$\leq n / b^{\log_b n - 1} + b/(b-1)$$

$$= n/(n/b) + b/(b-1) = b + b/(b-1) = O(1)$$

Recursion Tree

Recursion Tree of $T(n)=a\,T(\lceil n/b \rceil)+f(n)$

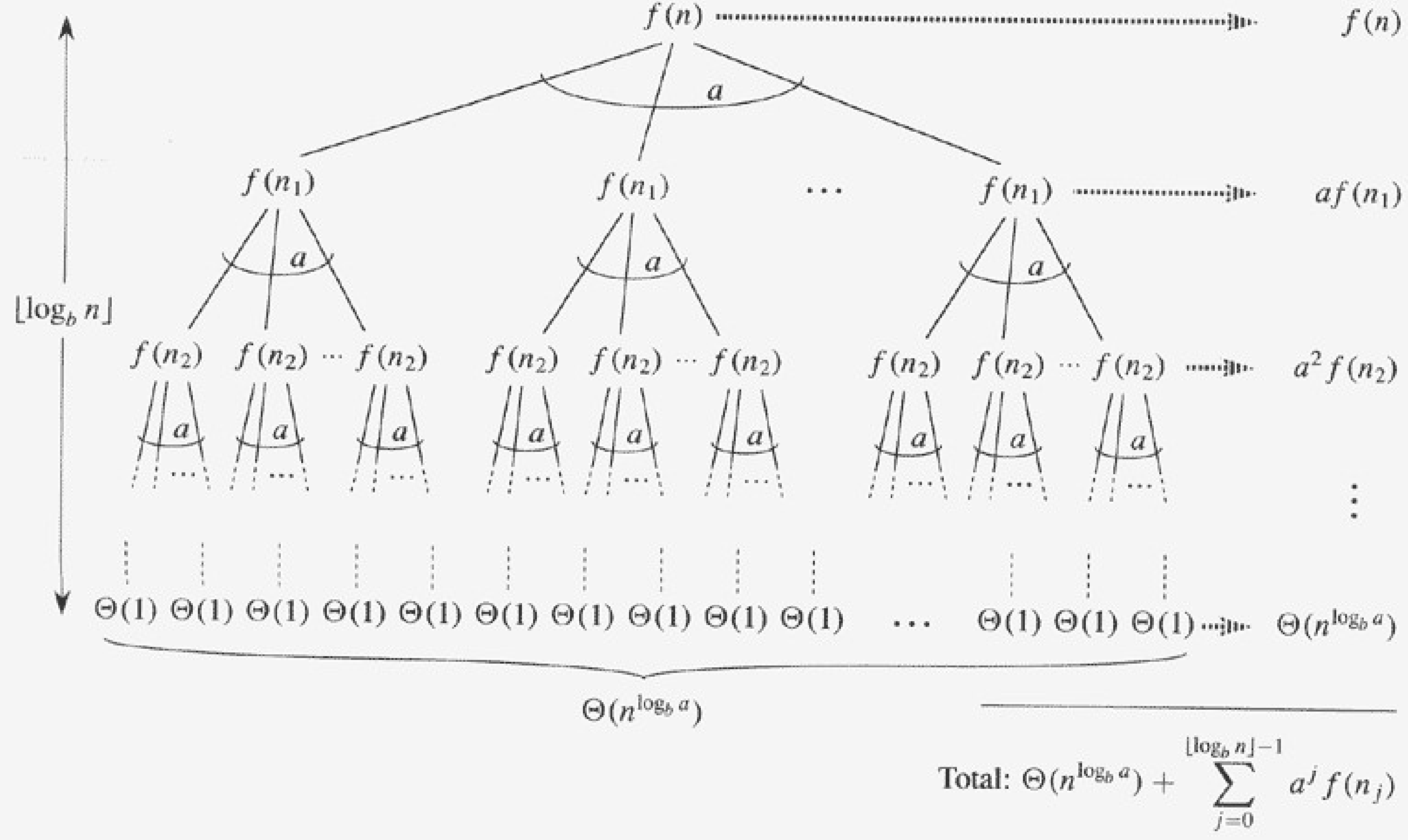


Figure 4.4 The recursion tree generated by $T(n) = aT(\lceil n/b \rceil) + f(n)$. The recursive argument n_j is given by equation (4.12).

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.

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

The Simple Format of Master Theorem

- $T(n)=aT(n/b)+cn^k$, with a, b, c, k are positive constants, and $a\geq 1$ and $b\geq 2$,
- $$T(n) = \begin{cases} O(n^{\log_b a}), & \text{if } a>b^k. \\ O(n^k \log n), & \text{if } a=b^k. \\ O(n^k), & \text{if } a<b^k. \end{cases}$$

Exercise (1)

Give asymptotic upper and lower bound for $T(n)=2T(n/4) + n^{0.5}$

Using the master theorem, a=2, b=4,

$n^{\log_b a} = \boldsymbol{n^{0.5}}$ and f (n) = $\boldsymbol{n^{0.5}}$. = $\Theta(\boldsymbol{n^{0.5}})$

Case 2 applies,

Therefore, T (n) = Θ ($\boldsymbol{n^{0.5}}$ lg n).

Exercise (2)

Give asymptotic upper and lower bound for $T(n)=7T(n/2) + n^2$

Using the master theorem, a=7, b=2,

$n^{\log_b a} = n^{\log_2 7}$

f (n) = $n^2 = O(n^{\log_2 7 - \varepsilon})$ for some constant $\varepsilon > 0$ due to
$2 < \lg 7 < 3$,

Case 1 applies,

Therefore, T (n) = $\Theta(n^{\log_2 7})$.

Exercise (3)

Give asymptotic upper and lower bound for $T(n)=7T(n/3) + n^2$

Using the master theorem, a=7, b=3, $n^{\log_b a} = n^{\log_3 7}$

$f(n) = n^2 = \Omega(n^{\log_3 7 + \varepsilon})$ for some constant $\varepsilon > 0$

Check if $a\, f(n/b) <= c\, f(n)$ for constant $c<1$,

$a(n/b)^2 = (7/9)\, n^2$

We can set $c=7/9 <1$, Case 3 applies,

Therefore, $T(n) = \Theta(n^2)$.

Exercise (4)

Give asymptotic upper and lower bound for $T(n)=16T(n/4)+n^2$

Using the master theorem, a=16, b=4, $n^{\log_b a} = n^{\log_4 16} = n^2$

f (n) = $n^2 = \Theta(n^2)$

Case 2 applies,

Therefore, T (n) = $\Theta(n^2 \lg n)$.

Exercise (5)

Give asymptotic upper and lower bound for $T(n)=T(n^{0.5}) + 1$

The easy way to do this is with a change of variables.

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

$T(2^m) = T(2^{m/2}) + 1$, So $S(m) = S(m/2)+1$,

Using the master theorem, a=1, b=2. $n^{\log_b a} = 1$ and f (n) = 1.

Case 2 applies and S(m) = Θ(lg m).

Therefore, T (n) = Θ (lg lg n).