Appendix B
Solving Recurrence Equations

- The analysis of recursive algorithms is not as straightforward as it is for iterative algorithms.
- it is not difficult to represent the time complexity of a recursive algorithm by a recurrence equation.
- The recurrence equation must then be solved to determine the time complexity.
- We discuss techniques for solving such equations and for using the solutions in the analysis of recursive algorithms.

B.1 Solving Recurrences Using Induction

Algorithm B.1 Factorial

- Problem: Determine ***n!*** *when n ≥ 1.,* 0! = 1
- Inputs: a nonnegative integer *n.*
- Outputs: *n!*.

```
int fact (int n )
{
   if ( n == 0 )
      return 1;
   else
      return n * fact ( n – 1 );
}
```

- t_n : the number of multiplications for a given value of *n*

$t_n = t_{n-1} + 1$, $t_0 = 0$ (*no multiplication are done when n=0*)

- Such equation is called *recurrence equation* because the value of the function at *n is* given in terms of the value of the function at a smaller value of *n.*

$t_1 = 1,\quad t_2 = t_1 + 1 = 2,\ t_3 = 3,$

- An inspection of the values just computed indicates that

$$t_n = n$$

Proof by induction

Example B.1

- Consider the recurrence

  $t_n = t_{n/2}+1$ *for* $n>1$, *n is a power of 2,* $t_1=1$

- We use induction to prove $t_n=\log n + 1$

Induction base: $n=1$, $t_1=\log 1 + 1 = 0 + 1 = 1$

Induction step: Assume, for an arbitrary $n > 0$ *and* n a power of *2,* that $t_n=\log n + 1$

Because the recurrence is only for powers of 2, the next value to consider after n is $2n$.

We need show that $t_{2n}= \log 2n +1$

$t_{2n}= t_n+1 = \log n + 1 +1= \log n + \log 2 +1= \log 2n +1$

Example B.2

- Consider the recurrence

$t_n = 7\, t_{n/2}$ *for n>1, n is a power of 2*

$t_1 = 1$

$t_1 = 7^0$,

$t_2 = 7\, t_{2/2} = 7\, t_1 = 7$, $t_4 = 7\, t_{4/2} = 7\, t_2 = \mathbf{7^2}$,

$t_8 = = 7\, t_{8/2} = 7\, t_4 = \mathbf{7^3}$,

$t_n =$ **?**

$t_n = 7^{\log n}$

Use induction to prove that this is correct.

Induction base: For $n = 1$,

$t_1=1, \quad 7^{\log 1} = 7^{0} = 1$

Induction hypothesis: Assume, for an arbitrary $n > 0$ *and n a power of 2, that* $t_n=7^{\log n}$

We need to show that $t_{2n}=7^{\log 2n}$**?**

$t_{2n}=7\, t_n$

$= 7 \times 7^{\log n} = 7^1 \times 7^{\log n}$

$= 7^{\log 2} \times 7^{\log n}$

$= 7^{\log 2 + \log n} = 7^{\log 2n}$

Example:

Consider the recurrence

$t_n = 2t_{n/2} + n + 1$ *for n>1, n is a power of 2,* $t_1 = 0$

There is no obvious candidate solution suggested by these values. As mentioned earlier, induction can only verify that a solution is correct. Because we have no candidate solution, we cannot use induction.

B.2 Solving Recurrences Using the Characteristic Equation

B.2.1 Homogeneous Linear Recurrences

Definition

- A recurrence of the form

  $a_0 t_n + a_1 t_{n-1} + \ldots + a_k t_{n-k} = 0$

  where k and the a_i terms are constants, is called a *homogeneous linear recurrence equation* with constant coefficients*:*

  - Linear : every term t_i *appears* only to the first power and : there be no terms $t_{c(n-i)}$, where *c* is a positive constant other than 1. For example, there may not be terms such as $t_{n/2}$, $t_{3(n-4)}$, etc.
  - Homogeneous: because the linear combination of the terms =0.

Homogeneous Linear Recurrences

$$
\begin{aligned}
7t_n - 3t_{n-1} &= 0 \\
6t_n - 5t_{n-1} + 8t_{n-2} &= 0 \\
8t_n - 4t_{n-3} &= 0
\end{aligned}
$$

Example B.4

The following is homogeneous linear recurrence equations with constant coefficients:

$$6\, t_n - 5\, t_{n-1} + 8\, t_{n-2} = 0$$

Example B.5

The Fibonacci sequence is defined as follows:

$$t_n = t_{n-1} + t_{n-2} \qquad t_0 = 0, \;\; t_1 = 1$$

$$\Rightarrow t_n - t_{n-1} - t_{n-2} = 0$$

which shows that the Fibonacci sequence is defined by a homogeneous linear recurrence.

Example B.6

- Suppose we have the recurrence

  $t_n - 5t_{n-1} + 6\, t_{n-2} = 0$ *for* $n>1$, $t_0 = 0$, $t_1 = 1$

- Notice that if we set $\boxed{t_n = r^n}$

  $r^n - 5\, r^{n-1} + 6\, r^{n-2} = 0$

  $r^{n-2}(r^2 - 5\, r^1 + 6) = 0$

  $r^{n-2}(r-2)(r-3) = 0$

- *Solution* $r = 0$ *or* $r = 2$ *or* $r = 3$

  $t_n = 0$ *or* $t_n = 2^n$ *or* $t_n = 3^n$

- We verify that these are solutions

- We verify this for 3^n by substituting it into the left side of the recurrence, as follows:
- $3^n - 5(3^{n-1}) + 6(3^{n-2})$

  $= 3^n - 5(3^{n-1}) + 2(3^{n-1})$

  $= 3^n - 3(3^{n-1}) = 0$

- The solution is

  $t_n = c_1 3^n + c_2 2^n$

  To find the constants

  $t_0 = 0 = c_1 + c_2$

  $t_1 = 1 = 3c_1 + 2c_2$

  $c_1 = 1, c_2 = -1$

  $t_n = 3^n - 2^n$

Definition

- The *characteristic equation for the homogeneous linear recurrence equation with constant* coefficients

$$a_0 t_n + a_1 t_{n-1} + \ldots + a_k t_{n-k} = 0$$

is defined as

$$a_0 r^k + a_1 r^{k-1} + \ldots + a_k r^0 = 0$$

Example B.7

- The characteristic equation for the recurrence

$5t_n - 7t_{n-1} + 6\ t_{n-2} = 0$

is

$5 r^2 - 7 r^1 + 6 = 0$

Theorem B.1

- Consider *the homogeneous linear recurrence equation with constant* coefficients

  $a_0 t_n + a_1 t_{n-1} +\ldots+ a_k t_{n-k} = 0$

  If its characteristic equation

  $a_0 r^k + a_1 r^{k-1} +\ldots+ a_k r^0 = 0$

  has *k* distinct solutions $\boxed{r_1\ r_2 \ldots r_k}$, then the only solutions to the recurrence are

$$\boxed{t_n = c_1 r_1^n + c_2 r_2^n + \ldots + c_k r_k^n}$$

  where the c_i terms are constants.

- The constants are determined by the initial conditions.

Example B.8

- We solve the recurrence

  $t_n - 3\,t_{n-1} - 4\;t_{n-2} = 0\ ,\ t_0 = 0,\ t_1 = 1$

  The characteristic equation

  $r^2 - 3\,r\ \ - 4 = 0$

  (r-4)(r+1)=0

- The roots are $r = 4$ *and* $r = -1.$
- Apply Theorem B.1 to get the general solution to the recurrence:

  $t_n = c_1\ 4^n + c_2\,(-1)^n$

- Determine the values of the constants ?

$c_1 = 1/5\ ,\ c_2 = -1/5$

- See **Example B.9,** the Fibonacci sequence:

$$t_n = \frac{\left[\left(1+\sqrt{5}\right)/2\right]^n - \left[\left(1-\sqrt{5}\right)/2\right]^n}{\sqrt{5}}.$$

- Theorem B.1 requires that all *k roots of the characteristic equation be* ***distinct****. The theorem does not* allow a characteristic equation of the following form:

$$(r-1)(r-2)^3=0$$

multiplicity 3

- *2 is called a root of multiplicity 3 of the equation.*

Theorem B.2

- Let ***r*** *be a root of multiplicity m of the characteristic equation for a homogeneous linear recurrence* with constant coefficients. Then

$$\boxed{t_n = r^n,\ t_n = n r^n,\ t_n = n^2 r^n,\ \ldots,\ t_n = n^{m-1} r^n}$$

are all solutions to the recurrence.

Example B.10

- We solve the recurrence

  $t_n - 7\,t_{n-1} + 15\;t_{n-2} - 9\,t_{n-3} = 0$, *for* $n>2$, $t_0 = 0$, $t_1 = 1$, $t_2 = 2$

  $r^3 - 7\,r^2 + 15\,r - 9 = 0$

  $(r-1)(r-3)^2 = 0$

- The roots are $r = 1$ *and* $r = 3$, of multiplicity *2.*
- Apply Theorem B.2 to get the solution to the recurrence:

  $t_n = c_1\;1^n + c_2\,3^n + c_3\,n\,3^n$

- Solving this system yields $c_1 = -1$, $c_2 = 1$, $c_3 = 1/3$
- **See Example B.11**

B.2.2 Nonhomogeneous Linear Recurrences

Definition

- A recurrence of the form

$$\boxed{a_0 t_n + a_1 t_{n-1} + \ldots + a_k t_{n-k} = \boldsymbol{f(n)}}$$

where k and the a_i terms are constants and $f(n)$ is a function other than the zero function, is called a **non**homogeneous linear recurrence equation with constant coefficients.

- We develop a method for solving the common special case

  $a_0 t_n + a_1 t_{n-1} + \ldots + a_k t_{n-k} = b^n p(n)$

  b : constant

  p (*n*) : polynomial in *n.*

Example B.12

- The recurrence

  $t_n - 3\, t_{n-1} = 4^n$

  $k = 1,\ b = 4,$ and $p\,(n) = 1.$

Example B.13

- The recurrence

  $t_n - 3\, t_{n-1} = 4^n\ (8n+7)$

  $k = 1,\ b = 4,\quad$ *and* $p\,(n) = 8n + 7.$

Example B.14

- We solve the recurrence

  $t_n - 3\, t_{n-1} = 4^n$, $t_0 = 0$, $t_1 = 4$

  Replace *n with n − 1*

  $t_{n-1} - 3\, t_{n-2} = 4^{n-1}$ ……. 1

  Divide the original recurrence by 4

  $t_n /4 - 3\, t_{n-1} /4 = 4^{n-1}$ …… 2

  Subtracting 1 from 2

  $t_n /4 - 7\, t_{n-1} /4 + 3\, t_{n-2} = 0$

- We can multiply by 4

$t_n - 7\, t_{n-1} + 12\, t_{n-2} = 0$

- This is a homogeneous linear recurrence equation, which means that it can be solved by applying Theorem B.1. That is, we solve the characteristic equation

$$r^2 - 7r + 12 = (r-3)(r-4) = 0,$$

obtain the general solution

$$t_n = c_1 3^n + c_2 4^n,$$

and use the initial conditions $t_0 = 0$ and $t_1 = 4$ to determine the particular solution:

$$t_n = 4^{n+1} - 4(3^n).$$

Theorem B.3

- A nonhomogeneous linear recurrence of the form

  $a_0 t_n + a_1 t_{n-1} + \ldots + a_k t_{n-k} = b^n p(n)$

  can be transformed into a homogeneous linear recurrence that has the characteristic equation

  $$\boxed{(a_0 r^k + a_1 r^{k-1} + \ldots + a_k)(r-b)^{d+1} = 0}$$

  where ***d*** is the **degree** *of* $p(n)$.
- If there is more than one term like $b^n p(n)$ on the right side, each one contributes a term to the characteristic equation.

Example B.15

Solve the recurrence

- $t_n - 3\, t_{n-1} = 4^n\, (2n+1\,)$ for n>1
- $t_0 = 0\,,\ t_1 = 12$

1. Obtain the charac. equation for the corresponding homo. equation

   r-3=0

2. Obtain a term from the nonhomo. part of the recurrence:

   $(r\text{-}b)^{\,d+1} = (r\text{-}4)^{\,1+1} = (r\text{-}4)^2$

3. From Theorem B.3: the characteristic equation (r-3) $(r\text{-}4)^2$ =0

   The roots are *r = 3 and r = 4, and the root r = 4 has multiplicity 2.*

   $t_n = c_1\, 3^{\,n} + c_2\, 4^{\,n} + c_3\, n\, 4^{\,n}$

$t_n = c_1 3^n + c_2 4^n + c_3 n 4^n$

- Substituting n=2 (the orginal recurrence equation):

$t_2 - 3 t_1 = 4^2 (2 \times 2 + 1)$

$t_2 = 3 t_1 + 80 = 116 = c_1 3^2 + c_2 4^2 + c_3 2 4^2$

Exercise

you are asked to determine the values c_1, c_2, *and* c_3

$t_n = 20 (3^n) - 20 (4^n) + 8 n 4^n$

Example B.16

Solve the recurrence

t_n - t_{n-1} = (n-1) for n>0

$t_0 = 0$??

Example B.17

- We solve the recurrence

  $t_n - 2\, t_{n-1} = n + 2^n$ for n>1

  $t_1 = 0$

  Determine the characteristic equation for the corresponding homogeneous recurrence

  $t_n - 2\, t_{n-1} = 0 \Rightarrow r-2=0$

  This is a case in which there are two terms on the right

n= 1^n (n),	$2^n = 2^n\,(n^0)$
$(r-1)^{1+1} = (r-1)^2$,	$(r-2)^{0+1} = r-2$

  Apply Theorem B.3 to obtain the characteristic equation from all the terms:

  $(r-2)\,(r-1)^2\,(r-2) = (r-2)^2\,(r-1)^2$

B.2.3 Change of Variables (Domain Transformations)

Example B.18

- We solve the recurrence$T(n) = T(n/2) + 1$ for $n>1$ and n power of 2

  $T(1) = 1$

- We can transform it into a recurrence that is in that form as follows. First, set **$n=2^k$**
- $T(2^k) = T(2^{k-1}) + 1$
- Next, set **$t_k = T(2^k)$**

  $t_k = t_{k-1} + 1$

  $t_k - t_{k-1} = 1$, $p(n)=1$, $b=1$, $p(n)$ is of degree 0

  $(r-1)\,(r-1)^{0+1} = 0$

  $(r-1)^2 = 0$

- Therefore, applying theorem B.3, we can determine its general solution to be

  $t_k = c_1 + c_2 k$

- Substitute $T(2^k)$ *for* t_k in the general solution

  $T(2^k) = c_1 + c_2 k$

  $T(n) = c_1 + c_2 \log n$

  $T(2) = T(1) + 1 = 1 + 1 = c_1 + c_2 \log 2 = c_1 + c_2 \Rightarrow 2 = c_1 + c_2$

  $T(1) = 1 = c_1 + c_2 \log 1 = c_1$

  $c_1 = 1, \; c_2 = 1$

  $T(n) = 1 + \log n$

Example B.19

$T(n) = 2\,T(n/2) + n - 1$ for $n > 1$ and n power of 2

$T(1) = 0$

$T(n) = 2\,T(n/2) + n-1$ for $n>1$ and n power of 2

$T(1) = 0$

- First, set $n=2^k$
- $T(2^k) = 2T(2^{k-1}) + 2^k - 1$
- Next, set $t_k = T(2^k)$

$t_k = 2t_{k-1} + 2^k - 1$

$\boldsymbol{t_k - 2t_{k-1} = 2^k - 1,}$

$t_k - 2t_{k-1} = 2^k - 1,$

2^k : b=2, p(n) =1 is of degree 0

1 : b=1, p(n) =-1 is of degree 0

$(r-2)(r-2)^{0+1} (r-1)^{0+1} = 0$

$t_k = c_1 + c_2 2^k + c_3 k 2^k$

$T(2^k) = c_1 + c_2 2^k + c_3 k 2^k$

$T(n) = c_1 + c_2 n + c_3 n \log n$

Excercise

- solve the recurrence eq

$T(n) = 7\,T(n/2) + 18\,(n/2)^2$ for $n>1$ and n power of 2, $T(1) = 0$

Example B.20

- We solve the recurrence

$T(n) = 7\,T(n/2) + 18\,(n/2)^2$ for $n>1$ and n power of 2, $T(1) = 0$

- First, set $n=2^k$
- $T(2^k) = 7T(2^{k-1}) + 18(\,2^{k-1})^2$
- Next, set $t_k = T(2^k)$

$t_k = 7t_{k-1} + 18\ 2^{2k-2}$

$\boldsymbol{t_k - 7t_{k-1} = 18\ 2^{2k-2} \Rightarrow t_k - 7t_{k-1} = 9/2\ 2^{2k}}$

$\boldsymbol{t_k - 7t_{k-1} = 9/2\ 4^k}$

4^k *:* $b=4$, $p(n) = 9/2$ is of degree 0

$(r-7)(r-4)^{0+1} = 0$

$T(2^k) = c_1\, 7^k + c_2\, 4^k$

$T(n) = c_1\, 7^{\log n} + c_2\, 4^{\log n}$

$T(n) = c_1\, n^{\log 7} + c_2\, n^2$

B.3 Solving Recurrences by Substitution

- The *method* of *substitution* if you cannot obtain a solution using the methods in the last two sections.

Example B.21

- We solve the recurrence

  $t_n = t_{n-1} + n$, for $n>1$, $t_1 = 1$

- In a sense, substitution is the opposite of induction. That is, we start at *n and work backward:*

$$t_n = t_{n-1} + n$$
$$t_{n-1} = t_{n-2} + n-1$$
$$t_{n-2} = t_{n-3} + n-2$$
$$\ldots\ldots\ldots$$
$$t_2 = t_1 + 2$$
$$t_1 = 1$$

→ substitute each equality into the previous one, as follows: →

$$t_n = t_{n-1} + n$$
$$t_n = t_{n-2} + (n-1) + n$$
$$= t_{n-3} + (n-2) + (n-1) + n$$
$$\ldots\ldots\ldots$$
$$= t_1 + 2 + \ldots + n$$
$$= 1 + 2 + \ldots + n$$
$$= n(n+1)/2$$

Example B.22

- We solve the recurrence

  $t_n = t_{n-1} + 2/n$, for n>1 , $t_1 = 0$

$$
\begin{aligned}
t_n &= t_{n-1} + \frac{2}{n} \\
t_{n-1} &= t_{n-2} + \frac{2}{n-1} \\
t_{n-2} &= t_{n-3} + \frac{2}{n-2} \\
&\vdots \\
t_2 &= t_1 + \frac{2}{2} \\
t_1 &= 0.
\end{aligned}
$$

$$
\begin{aligned}
t_n &= t_{n-1} + \frac{2}{n} \\
&= t_{n-2} + \frac{2}{n-1} + \frac{2}{n} \\
&= t_{n-3} + \frac{2}{n-2} + \frac{2}{n-1} + \frac{2}{n} \\
&\vdots \\
&= t_1 + \frac{2}{2} + \cdots + \frac{2}{n-2} + \frac{2}{n-1} + \frac{2}{n} \\
&= 0 + \frac{2}{2} + \cdots + \frac{2}{n-2} + \frac{2}{n-1} + \frac{2}{n} \\
&= 2\sum_{i=2}^{n} \frac{1}{i} \approx 2 \ln n
\end{aligned}
$$

For large n, see appendix A

B.4 Extending Results *to n in* General

- Recursive algorithms: we can determine the exact time complexity only when *n* is a power of some base *b, where b is a positive constant. Often the base b is 2.*

- It seems that a result that holds for *n a power of b should approximately hold for n in general.*

- *For example, if for* some algorithm we establish that

  T(n)=2 n log n , for *n a power of 2,*

  it seems that for n in general we should be able to conclude that $T(n) \in \theta(n \log n)$

Definition

- A complexity function $f(n)$ *is called* *strictly increasing* *if*

  $n_1 > n_2 \Rightarrow \quad f(n_1) > f(n_2)$

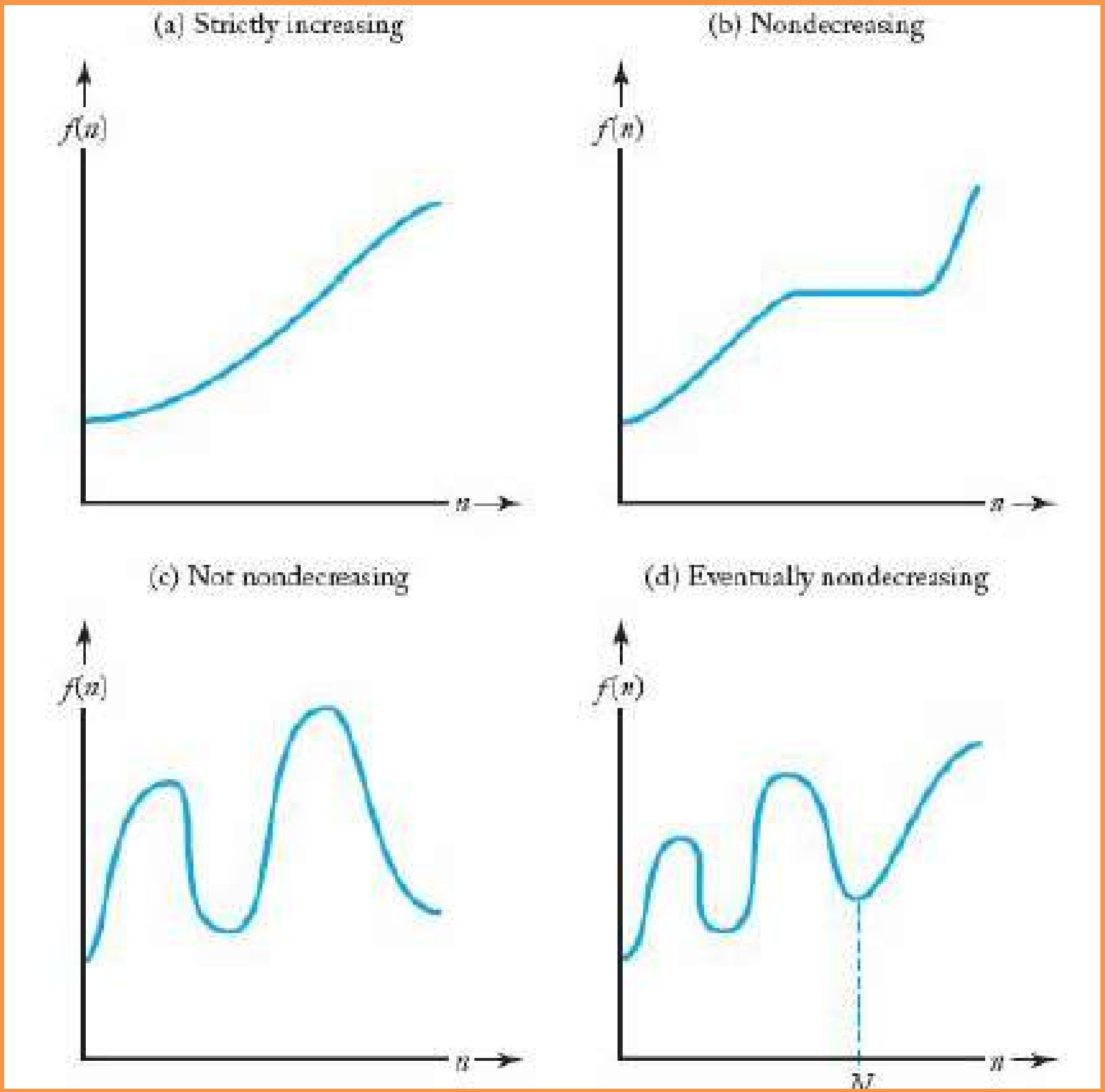
Example

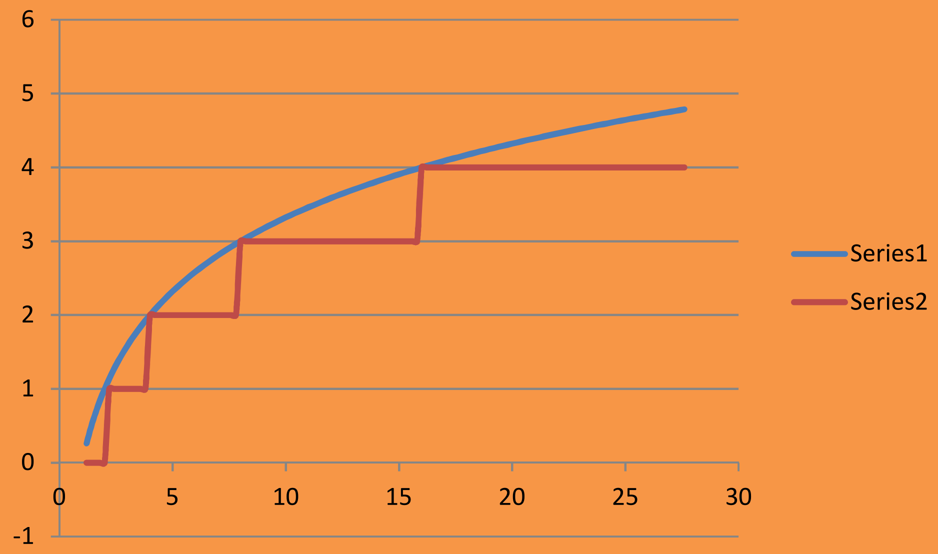
- *lg n, n, n lg n, n^2, and* $\boldsymbol{2^n}$ are all strictly increasing as long as n is nonnegative.

Definition

- A complexity function $f(n)$ *is called* *nondecreasing* *if*

  $n_1 > n_2 \Rightarrow f(n_1) >= f(n_2)$


(a) Strictly increasing
(b) Nondecreasing
(c) Not nondecreasing
(d) Eventually nondecreasing
f(n)
n
M



Log n, floor(log n))

- The time (or memory) complexities of most algorithms are nondecreasing.

Definition

- A complexity function $f(n)$ *is called eventually nondecreasing if there exists an N such that*

  $n_1 > n_2 > N \Rightarrow$ *then* $f(n_1) >= f(n_2)$

- Any nondecreasing function is eventually nondecreasing

Definition

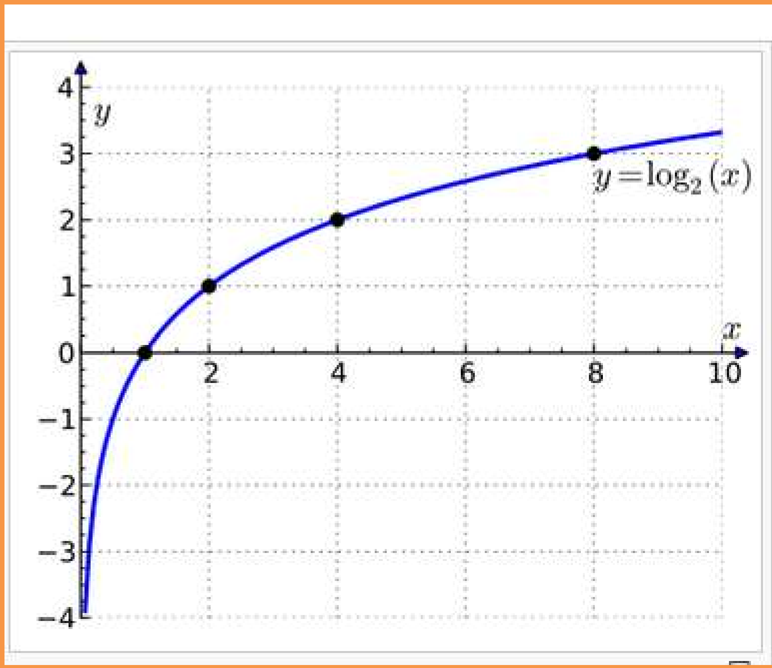
- A complexity function $f(n)$ *is called smooth if*
  - $f(n)$ *eventually nondecreasing and*
  - *and if* $f(2n) \in \theta(f(n))$

Example B.23

- The functions lg n, n, $n \lg n$, and n^k, where $k \geq 0$, are all smooth.

Example

- lg n *is* eventually nondecreasing.
- As to the second condition, we have $\log(2n) = \log 2 + \log n \in \theta(\log n)$



Example B.24

- The function 2^n ***is not smooth,***
- By property 4: $2^n \in o(4^n)$

  therefore, $2^{2n} = 4^n \notin \theta(2^n)$

Theorem B.4

- Let *b ≥ 2 be an integer, let f(n) be a smooth complexity function, and let T (n) be an eventually* nondecreasing complexity function.

If $T(n) \in \theta(f(n))$ for n a power of b

Then $T(n) \in \theta(f(n))$ for any n

Furthermore, the same implication holds if Θ is replaced by *"*big *O," Ω, or "small o."*

Example B.25

- Suppose for some complexity function we establish that $T(n)=T(\lfloor n/2 \rfloor) +1, n>1, T(1)=1$
- When *n is a power of 2, we* have

  $T(n)=1+ \log n \in \theta(\log n)$
- Because *lg n is smooth, we need only show that T (n) is eventually nondecreasing in order to apply* Theorem B.4 to conclude that $T(n) \in \theta(\log n)$

- $W(n)=W(\lfloor n/2 \rfloor) + W(\lceil n/2 \rceil) + n-1$

- $T(n)$ is eventually nondecreasing
- We use induction to establish for $n \geq 2$ *that*

$$\boxed{\textit{if } 1 \leq k < n, \textit{ then} \text{ that } T(k) \leq T(n)}$$

- Induction base: For $n = 2$, *we must show that* $T(1) \leq T(2)$

  $T(1)=1$

  $T(2)= T(\lfloor 2/2 \rfloor) +1=T(1)+1=2$

  Therefore, $T(1) \leq T(2)$

- Induction hypothesis: Let $n >= 2$.
- Assume for all $k \le n$ *that* $T(k) \le T(n)$
- Induction step: we need only show that $T(n) \le T(n+1)$
- It is not hard to see that if $n \ge 1$, *then* $\lfloor n/2 \rfloor \le \lfloor (n+1)/2 \rfloor$

Therefore, by the induction hypothesis,

$$T(\lfloor n/2 \rfloor) \le T(\lfloor (n+1)/2 \rfloor)$$

Using the recurrence, we have

$$T(n) = T(\lfloor n/2 \rfloor) + 1 \le T(\lfloor (n+1)/2 \rfloor) + 1 = T(n+1)$$

Theorem B.5

Suppose a complexity function $T(n)$ is eventually nondecreasing and satisfies

$$\boxed{\begin{array}{ll} T(n) = aT\left(\frac{n}{b}\right) + cn^k & \text{for } n > 1,\ n \text{ a power of } b \\ T(1) = d & \end{array}}$$

where $b \geq 2$ and $k \geq 0$ are constant integers, and a, c, and d are constants such that $a > 0$, $c > 0$, and $d \geq 0$. Then

$$T(n) \in \begin{cases} \Theta\left(n^k\right) & \text{if } a < b^k \\ \Theta\left(n^k \lg n\right) & \text{if } a = b^k \\ \Theta\left(n^{\log_b a}\right) & \text{if } a > b^k. \end{cases} \qquad \text{(B.5)}$$

Furthermore, if, in the statement of the recurrence,

$$T(n) = aT\left(\frac{n}{b}\right) + cn^k$$

is replaced by

$$T(n) \leq aT\left(\frac{n}{b}\right) + cn^k \qquad \text{or} \qquad T(n) \geq aT\left(\frac{n}{b}\right) + cn^k,$$

then Result B.5 holds with "big *O" or Ω, respectively, replacing Θ.*

Example B.26

- Suppose that $T(n)$ *is eventually nondecreasing* and satisfies

  $T(n) = 8T(n/4) + 5n^2$ for $n>1$ a power of 4

  $T(1)=3$

  By Theorem B.5, a=8, b=4,k=2,because $8 < 4^2$, $T(n) \in \theta(n^2)$

Example B.27

- Suppose that $T(n)$ is *eventually nondecreasing* and satisfies

  $T(n) = 9\,T(n/3) + 5n^1$ for n>1 a power of 3

  $T(1)=7$

  By Theorem B.5, because $9 > 3^1$, $T(n) \in \theta(n^{\log_3 9}) = \theta(n^2)$

Theorem B.6

- Suppose that a complexity function $T(n)$ *is eventually nondecreasing and satisfies*

$$T(n) = aT\left(\frac{n}{b}\right) + cn^k \qquad \text{for } n > 2,\ n \text{ a power of } b$$
$$\boxed{T(s) = d}$$

- where *s is* a constant that is a power *of b, b ≥ 2 and k ≥ 0* are constant integers, and *a, c,* and d are constants such that *a > 0, c > 0, and d ≥ 0.* Then the results in Theorem B.5 still hold.

Example B.28

- Suppose that *T (n) is eventually nondecreasing and satisfies*

  $T(n) = 8\,T(n/2) + 5n^3$ for n>64 a power of 2

  $T(64) = 200$

  By Theorem B.6, because b=2, a=8 = 2^3,**k=3,** $b^k = 2^3$**,**

  $T(n) \in \theta(n^3 \log n)$