Problems with solutions in the Analysis of Algorithms

Draft date April 23, 2009

Chapter 1

Notations: Θ, O, Ω, o, and ω

The functions we consider are assumed to have positive real domains and real codomains (ranges), unless specified otherwise. Furthermore, the functions are *asymptotically positive*: $f(n)$ is asymptotically positive iff $\exists n_0 : \forall n \geq n_0, f(n) > 0$.

Basic definitions:

$$\Theta(g(n)) = \{f(n) \mid \exists c_1, c_2 > 0, \exists n_0 : \ \forall n \geq n_0, \ 0 \leq c_1.g(n) \leq f(n) \leq c_2.g(n)\} \quad (1.1)$$

$$O(g(n)) = \{f(n) \mid \exists c > 0, \exists n_0 : \ \forall n \geq n_0, \ 0 \leq f(n) \leq c_2.g(n)\} \quad (1.2)$$

$$\Omega(g(n)) = \{f(n) \mid \exists c > 0, \exists n_0 : \ \forall n \geq n_0, \ 0 \leq c.g(n) \leq f(n)\} \quad (1.3)$$

$$o(g(n)) = \{f(n) \mid \forall c > 0, \exists n_0 : \ \forall n \geq n_0, \ 0 \leq f(n) < c.g(n)\} \quad (1.4)$$

$$\omega(g(n)) = \{f(n) \mid \forall c > 0, \exists n_0 : \ \forall n \geq n_0, \ 0 \leq c.g(n) < f(n)\} \quad (1.5)$$

1.4 is equivalent to:

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0 \quad (1.6)$$

but only in case the limit exists. 1.5 is equivalent to:

$$\lim_{n \to \infty} \frac{g(n)}{f(n)} = 0 \quad (1.7)$$

but only in case the limit exists.

It is universally accepted to write "$\mathsf{f}(n) = \Theta(\mathsf{g}(n))$" instead of the formally correct "$\mathsf{f}(n) \in \Theta(\mathsf{g}(n))$".

Let us define the binary relations $\approx$, $\preceq$, $\prec$, $\succeq$, and $\succ$ over functions as follows. For any two functions $\mathsf{f}(n)$ and $\mathsf{g}(n)$, $n \in \mathbb{R}^+$, $\mathsf{f}(n), \mathsf{g}(n) \in \mathbb{R}$:

$$\mathsf{f}(n) \approx \mathsf{g}(n) \quad \Leftrightarrow \quad \mathsf{f}(n) = \Theta(\mathsf{g}(n)) \tag{1.8}$$

$$\mathsf{f}(n) \preceq \mathsf{g}(n) \quad \Leftrightarrow \quad \mathsf{f}(n) = \mathsf{O}(\mathsf{g}(n)) \tag{1.9}$$

$$\mathsf{f}(n) \prec \mathsf{g}(n) \quad \Leftrightarrow \quad \mathsf{f}(n) = \mathsf{o}(\mathsf{g}(n)) \tag{1.10}$$

$$\mathsf{f}(n) \succeq \mathsf{g}(n) \quad \Leftrightarrow \quad \mathsf{f}(n) = \Omega(\mathsf{g}(n)) \tag{1.11}$$

$$\mathsf{f}(n) \succ \mathsf{g}(n) \quad \Leftrightarrow \quad \mathsf{f}(n) = \omega(\mathsf{g}(n)) \tag{1.12}$$

When the relations do not hold we write $\mathsf{f}(n) \not\approx \mathsf{g}(n)$, $\mathsf{f}(n) \not\preceq \mathsf{g}(n)$, *etc.*

Properties of the relations:

1. Reflexivity: $\mathsf{f}(n) \approx \mathsf{f}(n)$, $\mathsf{f}(n) \preceq \mathsf{f}(n)$, $\mathsf{f}(n) \succeq \mathsf{f}(n)$.
2. Symmetry: $\mathsf{f}(n) \approx \mathsf{g}(n) \;\Leftrightarrow\; \mathsf{g}(n) \approx \mathsf{f}(n)$.
3. Transitivity:
   $\mathsf{f}(n) \approx \mathsf{g}(n)$ and $\mathsf{g}(n) \approx \mathsf{h}(n) \;\Rightarrow\; \mathsf{f}(n) \approx \mathsf{h}(n)$
   $\mathsf{f}(n) \preceq \mathsf{g}(n)$ and $\mathsf{g}(n) \preceq \mathsf{h}(n) \;\Rightarrow\; \mathsf{f}(n) \preceq \mathsf{h}(n)$
   $\mathsf{f}(n) \prec \mathsf{g}(n)$ and $\mathsf{g}(n) \prec \mathsf{h}(n) \;\Rightarrow\; \mathsf{f}(n) \prec \mathsf{h}(n)$
   $\mathsf{f}(n) \succeq \mathsf{g}(n)$ and $\mathsf{g}(n) \succeq \mathsf{h}(n) \;\Rightarrow\; \mathsf{f}(n) \succeq \mathsf{h}(n)$
   $\mathsf{f}(n) \succ \mathsf{g}(n)$ and $\mathsf{g}(n) \succ \mathsf{h}(n) \;\Rightarrow\; \mathsf{f}(n) \succ \mathsf{h}(n)$.
4. Transpose symmetry:
   $\mathsf{f}(n) \succeq \mathsf{g}(n) \;\Leftrightarrow\; \mathsf{g}(n) \preceq \mathsf{f}(n)$
   $\mathsf{f}(n) \succ \mathsf{g}(n) \;\Leftrightarrow\; \mathsf{g}(n) \prec \mathsf{f}(n)$.
5. $\mathsf{f}(n) \prec \mathsf{g}(n) \;\Rightarrow\; \mathsf{f}(n) \preceq \mathsf{g}(n)$
   $\mathsf{f}(n) \preceq \mathsf{g}(n) \;\not\Rightarrow\; \mathsf{f}(n) \prec \mathsf{g}(n)$
   $\mathsf{f}(n) \succ \mathsf{g}(n) \;\Rightarrow\; \mathsf{f}(n) \succeq \mathsf{g}(n)$
   $\mathsf{f}(n) \succeq \mathsf{g}(n) \;\not\Rightarrow\; \mathsf{f}(n) \succ \mathsf{g}(n)$

6. $f(n) \approx g(n) \Rightarrow f(n) \not\prec g(n)$
   $f(n) \approx g(n) \Rightarrow f(n) \not\succ g(n)$
   $f(n) \prec g(n) \Rightarrow f(n) \not\approx g(n)$
   $f(n) \prec g(n) \Rightarrow f(n) \not\succ g(n)$
   $f(n) \prec g(n) \Rightarrow f(n) \not\succeq g(n)$
   $f(n) \succ g(n) \Rightarrow f(n) \not\approx g(n)$
   $f(n) \succ g(n) \Rightarrow f(n) \not\prec g(n)$
   $f(n) \succ g(n) \Rightarrow f(n) \not\preceq g(n)$
7. $f(n) \approx g(n) \Leftrightarrow f(n) \preceq g(n)$ and $f(n) \succeq g(n)$
8. There do not exist functions $f(n)$ and $g(n)$, such that $f(n) \prec g(n)$ and $f(n) \succ g(n)$
9. Let $f(n) = f_1(n) + f_2(n) + f_3(n) + \ldots + f_k(n)$. Let

   $$\begin{aligned} &f_1(n) \succ f_2(n) \\ &f_1(n) \succ f_3(n) \\ &\quad \ldots \\ &f_1(n) \succ f_k(n) \end{aligned}$$

   Then $f(n) \approx f_1(n)$.
10. Let $f(n) = f_1(n) \times f_2(n) \times \ldots \times f_k(n)$. Let some of the $f_i(n)$ functions are constants. Without loss of generality, let those be the first m functions for some m such that $1 \leq m \leq k$. Namely, $f_1(n) = \text{const}$, $f_2(n) = \text{const}$, $\ldots$, $f_m(n) = \text{const}$. Then $f(n) \approx f_{m+1}(n) \times f_{m+2}(n) \times \ldots \times f_k(n)$.
11. Provided that $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ exists,

    $$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \text{const} \Leftrightarrow f(n) \approx g(n) \tag{1.13}$$

    However, without the provision the limit exists, it is the case that

    $$\begin{aligned} &\lim_{n\to\infty} \frac{f(n)}{g(n)} = \text{const} \Rightarrow f(n) \approx g(n) \\ &\lim_{n\to\infty} \frac{f(n)}{g(n)} = \text{const} \not\Leftarrow f(n) \approx g(n) \end{aligned}$$

    To see why, consider the example $f(n) = n^2$ and $g(n) = (2 + \sin(n))n^2$. Obviously $g(n)$ oscillates between n^2 and $3n^2$ and thus $f(n) \approx g(n)$ but it is not the case that $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \text{const}$ because the limit does not exist.

Problem 1 ([CLR00], pp. 24–25)**.** *Let* $f(n) = \frac{1}{2}n^2 - 3n$. *Prove that* $f(n) \approx n^2$.

Solution:
For a complete solution we have to show some concrete positive constants c_1 and c_2 and a concrete value n_0 for the variable, such that for all $n \geq n_0$,

$$0 \leq c_1.n^2 \leq \frac{1}{2}n^2 - 3n \leq c_2.n^2$$

Since $n > 0$ this is equivalent to (divide by n^2):

$$0 \leq c_1 \leq \frac{1}{2} - \frac{3}{n} \leq c_2$$

What we have here are in fact three inequalities:

$$0 \leq c_1 \tag{1.14}$$

$$c_1 \leq \frac{1}{2} - \frac{3}{n} \tag{1.15}$$

$$\frac{1}{2} - \frac{3}{n} \leq c_2 \tag{1.16}$$

(1.14) is trivial, any $c_1 > 0$ will do. To satisfy (1.16) we can pick $n_0' = 1$ and then any positive c_2 will do; say, $c_2 = 1$. The smallest integer value for n that makes the right-hand side of (1.15) positive is 7; the right-hand side becomes $\frac{1}{2} - \frac{3}{7} = \frac{7}{14} - \frac{6}{14} = \frac{1}{14}$. So, to saisfy (1.15) we pick $c_1 = \frac{1}{14}$ and $n_0'' = 7$. The overall n_0 is $n_0 = \max\{n_0', n_0''\} = 7$. The solution $n_0 = 7$, $c_1 = \frac{1}{14}$, $c_2 = 1$ is obviusly not unique. □

Problem 2. *Is it true that* $\frac{1}{1000}n^3 \preceq 1000n^2$*?*

Solution:
No. Assume the opposite. Then $\exists c > 0$ and $\exists n_0$, such that for all $n \geq n_0$:

$$\frac{1}{1000}n^3 \leq c.1000n^2$$

It follows that $\forall n \geq n_0$:

$$\frac{1}{1000}n \leq 1000.c \Leftrightarrow n \leq 1000000.c$$

That is clearly false. □

Problem 3. *Is it true that for any two functions, at least one of the five relations* $\approx$*,* $\preceq$*,* $\prec$*,* $\succeq$*, and* $\succ$ *holds between them?*

Solution:
No. Proof by demonstrating an counterexample ([CLR00, pp. 31]): let $f(n) = n$ and $g(n) = n^{1+\sin n}$. Since $g(n)$ oscillates between $n^0 = 1$ and n^2, it cannot be the case that $f(n) \approx g(n)$ or $f(n) \preceq g(n)$ or $f(n) \prec g(n)$ or $f(n) \succeq g(n)$ or $f(n) \succ g(n)$.

However, this argument from [CLR00] holds only when $n \in \mathbb{R}$. If $n \in \mathbb{N}^+$, we cannot use the function $g(n)$ directly, *i.e.* without extra arguments. Note that $\sin n$ reaches its extreme values -1 and 1 at $2k\pi + \frac{3\pi}{2}$ and $2k\pi + \frac{\pi}{2}$, respectively, for integer k. As these are irrational numbers, the integer n cannot be equal to any of them. So, it is no longer true that $g(n)$ oscillates between $n^0 = 1$ and n^2. If we insist on using $g(n)$ in our counterexample we have to argue, for instance, that:

- for infinitely many (positive) values of the integer variable, for some constant $\epsilon > 0$, it is the case that $g(n) \geq n^{1+\epsilon}$;
- for infinitely many (positive) values of the integer variable, for some constant $\sigma > 0$, it is the case that $g(n) \leq n^{1-\sigma}$.

An alternative is to use the function $g'(n) = n^{1+\sin(\pi n - \pi/2)}$ that indeed oscillates between $n^0 = 1$ and n^2 for integer n. Another alternative is to use

$$g''(n) = \begin{cases} n^2, & \text{if } n \text{ is even,} \\ 1, & \text{else.} \end{cases}$$

□

Problem 4. *Let $p(n)$ be any univariate polynomial of degree k, such that the coefficient in the higherst degree term is positive. Prove that $p(n) \approx n^k$.*

Solution:
$p(n) = a_k n^k + a_{k-1} n^{k-1} + \ldots + a_1 n + a_0$ with $a_k > 0$. We have to prove that there exist positive constants c_1 and c_2 and some n_0 such that for all $n \geq n_0$, $0 \leq c_1 n^k \leq p(n) \leq c_2 n^k$. Since the leftmost inequality is obvious, we have to prove that

$$c_1 n^k \leq a_k n^k + a_{k-1} n^{k-1} + a_{k-2} n^{k-2} \ldots + a_1 n + a_0 \leq c_2 n^k$$

For positive n we can divide by n^k, obtaining:

$$c_1 \leq a_k + \underbrace{\frac{a_{k-1}}{n} + \frac{a_{k-2}}{n^2} + \ldots + \frac{a_1}{n^{k-1}} + \frac{a_0}{n^k}}_{T} \leq c_2$$

Now it is obvious that any c_1 and c_2 such that $0 < c_1 < a_k$ and $c_2 > a_k$ are suitable because $\lim_{n\to\infty} T = 0$.

□

Problem 5. *Let $a \in \mathbb{R}$ and $b \in \mathbb{R}^+$. Prove that $(n+a)^b \approx n^b$*

Solution:
Note that this problem does not reduce to Problem 4 except in the special case when b is integer. We start with the following trivial observations:

$$n + a \leq n + |a| \leq 2n, \text{ provided that } n \geq |a|$$
$$n + a \geq n - |a| \geq \frac{n}{2}, \text{ provided that } \frac{n}{2} \geq |a|, \text{ that is, } n \geq 2|a|$$

It follows that:

$$\frac{1}{2} n \leq n + a \leq 2n, \text{ if } n \geq 2|a|$$

By raising to the b^{th} power we obtain:

$$\left(\frac{1}{2}\right)^b n^b \leq (n+a)^b \leq 2^b n^b$$

So we have a proof with $c_1 = \left(\frac{1}{2}\right)^b$, $c_2 = 2^b$, and $n_0 = \lceil 2|a| \rceil$. □

Problem 6. *Prove that for any two asymptotically positive functions* $f(n)$ *and* $g(n)$*, it is the case that* $\max(f(n), g(n)) \approx f(n) + g(n)$.

Solution:
We are asked to prove there exist positive constants c_1 and c_2 and a certain n_0, such that for all $n \geq n_0$:

$$0 \leq c_1(f(n) + g(n)) \leq \max(f(n), g(n)) \leq c_2(f(n) + g(n))$$

As $f(n)$ and $g(n)$ are asymptotically positive,

$$\exists n_0' : \forall n \geq n_0', f(n) > 0$$
$$\exists n_0'' : \forall n \geq n_0'', g(n) > 0$$

Let $n_0''' = \max\{n_0', n_0''\}$. Obviously,

$$0 \leq c_1(f(n) + g(n)) \text{ for } n \geq n_0''', \text{ if } c_1 > 0$$

It is also obvious that when $n \geq n_0'''$:

$$\frac{1}{2}f(n) + \frac{1}{2}g(n) \leq \max(f(n), g(n))$$
$$f(n) + g(n) \geq \max(f(n), g(n)),$$

which we can write as:

$$\frac{1}{2}(f(n) + g(n)) \leq \max(f(n), g(n)) \leq f(n) + g(n)$$

So we have a proof with $n_0 = n_0'''$, $c_1 = \frac{1}{2}$, and $c_2 = 1$. □

Problem 7. *Which of the following are true:*

$$2^{n+1} \approx 2^n$$
$$2^{2n} \approx 2^n$$

Solution:
$2^{n+1} \approx 2^n$ is true because $2^{n+1} = 2.2^n$ and for any constant c, $c.2^n \approx 2^n$. On the other hand, $2^{2n} \approx 2^n$ is not true. Assume the opposite. Then, having in mind that $2^{2n} = 2^n.2^n$, it is the case that for some constant c_2 and all $n \to +\infty$:

$$2^n.2^n \leq c_2.2^n \quad \Leftrightarrow \quad 2^n \leq c_2$$

That is clearly false. □

Problem 8. *Which of the following are true:*

$$\frac{1}{n^2} \prec \frac{1}{n} \tag{1.17}$$
$$2^{\frac{1}{n^2}} \prec 2^{\frac{1}{n}} \tag{1.18}$$

Solution:
(1.17) is true because

$$0 \leq \frac{1}{n^2} < c.\frac{1}{n} \quad \Leftrightarrow \quad 0 \leq \frac{1}{n} < c$$

is true for every positive constant c and sufficiently large n. (1.18), however, is not true. Assume the opposite. Then:

$$\forall c > 0, \exists n_0 : \forall n \geq n_0,\ 0 \leq 2^{\frac{1}{n^2}} < c.2^{\frac{1}{n}} \quad \Leftrightarrow \quad 0 \leq \frac{2^{\frac{1}{n^2}}}{2^{\frac{1}{n}}} < c \tag{1.19}$$

But

$$\lim_{n\to\infty}\left(\frac{2^{\frac{1}{n^2}}}{2^{\frac{1}{n}}}\right) = \lim_{n\to\infty}\left(2^{\frac{1}{n^2}-\frac{1}{n}}\right) = 1 \text{ because} \tag{1.20}$$

$$\lim_{n\to\infty}\left(\frac{1}{n^2}-\frac{1}{n}\right) = \lim_{n\to\infty}\left(\frac{1-n}{n^2}\right) = \lim_{n\to\infty}\left(\frac{\frac{1}{n}-1}{n}\right) = 0 \tag{1.21}$$

It follows that (1.19) is false. □

Problem 9. *Let a be a constant such that $a > 1$. Which of the following are true:*

$$f(n) \approx g(n) \ \Rightarrow a^{f(n)} \approx a^{g(n)} \tag{1.22}$$

$$f(n) \preceq g(n) \ \Rightarrow a^{f(n)} \preceq a^{g(n)} \tag{1.23}$$

$$f(n) \prec g(n) \ \Rightarrow a^{f(n)} \prec a^{g(n)} \tag{1.24}$$

for all asymptotically positive functions $f(n)$ and $g(n)$.

Solution:
(1.22) is not true – Problem 7 provides a counterexample since $2n \approx n$ and $2^{2n} \not\approx 2^n$. The same counterexample suffices to prove that (1.23) is not true – note that $2n \preceq n$ but $2^{2n} \not\preceq 2^n$.

Now consider (1.24).

case 1, $g(n)$ is increasing: True. Assume $f(n) \prec g(n)$. According to (1.4) on page 1,

$$\forall c > 0,\ \exists n_0 : \forall n \geq n_0,\ 0 \leq f(n) < c.g(n) \tag{1.25}$$

It follows that, for any $c > 0$ and all sufficiently large n,

$$a^{f(n)} < a^{c.g(n)} \tag{1.26}$$

We have to prove that:

$$\forall k > 0,\ \exists n_1 : \forall n \geq n_1,\ 0 \leq a^{f(n)} < k.a^{g(n)}$$

$0 \leq a^{f(n)}$ is trivially true so our task reduces to proving that for any $k > 0$ and all sufficiently large n,

$$a^{f(n)} < k.a^{g(n)} = a^{\log_a k}.a^{g(n)} = a^{\log_a k + g(n)}$$

Renaming $\log_a k$ to k_1, what we have to prove is that for any constant $k_1 > 0$ and sufficiently large n,

$$a^{f(n)} < a^{k_1 + g(n)} \tag{1.27}$$

Consider any fixed $k_1 > 0$. No matter how large k_1 is, for any positive constant $c_1 > 1$, for all sufficiently large n—provided that $g(n)$ is increasing—it is the case that:

$$k_1 + g(n) \leq c_1.g(n)$$

Therefore, under the same assumptions,

$$a^{k_1 + g(n)} \leq a^{c_1.g(n)} \tag{1.28}$$

From (1.27) and (1.28) it follows that what we have to prove is:

$$a^{f(n)} < a^{c_1.g(n)}, \text{ for any constant } c_1 > 1. \tag{1.29}$$

Compare (1.29) with (1.26), which is given. Since (1.26) holds for any positive c, certainly it holds with the additional restriction that the constant is greater than one, and thus we conclude that (1.29) is true.

case 2, $g(n)$ is not increasing: In this case (1.24) is not true. Consider Problem 8. As it is shown there, $\frac{1}{n^2} \prec \frac{1}{n}$ but $2^{\frac{1}{n^2}} \not\prec 2^{\frac{1}{n}}$. □

Problem 10. *Let a be a constant such that $a > 1$. Which of the following are true:*

$$a^{f(n)} \approx a^{g(n)} \Rightarrow f(n) \approx g(n) \tag{1.30}$$

$$a^{f(n)} \preceq a^{g(n)} \Rightarrow f(n) \preceq g(n) \tag{1.31}$$

$$a^{f(n)} \prec a^{g(n)} \Rightarrow f(n) \prec g(n) \tag{1.32}$$

for all asymptotically positive functions $f(n)$ and $g(n)$.

Solution:
(1.30) is true, if $g(n)$ is increasing. Suppose there exist positive constants c_1 and c_2 and some n_0 such that

$$0 \leq c_1.a^{g(n)} \leq a^{f(n)} \leq c_2.a^{g(n)}, \forall n \geq n_0$$

Since $a > 1$ and $f(n)$ and $g(n)$ are asymptotically positive, for all sufficiently large n, the exponents have strictly larger than one values. Therefore, we can take logarithm to base a (ignoring the leftmost inequality) to obtain:

$$\log_a c_1 + g(n) \leq f(n) \leq \log_a c_2 + g(n)$$

First note that for any constant k_1 such that $0 < k_1 < 1$, $k_1.g(n) \leq \log_a c_1 + g(n)$ for all sufficiently large n, regardless of whether the logarithm is positive or negative or zero. Then note that for any constant k_2 such that $k_2 > 1$, $\log_a c_2 + g(n) \leq k_2.g(n)$ for all sufficiently

large n, regardless of whether the logarithm is positive or negative or zero. Conclude there exists n_1, such that

$$k_1.g(n) \leq f(n) \leq k_2.g(n),\ \forall n \geq n_1$$

However, if $g(n)$ is not increasing, (1.30) is not true. We already showed (see (1.20)) that $\lim_{n\to\infty} \left(\frac{2^{\frac{1}{n^2}}}{2^{\frac{1}{n}}} \right) = 1$. According to (1.13), it follows that $2^{\frac{1}{n^2}} \approx 2^{\frac{1}{n}}$. However, $\frac{1}{n^2} \not\approx \frac{1}{n}$ (see (1.21)).

Consider (1.31). It is true if $g(n)$ is increasing and not true otherwise. If $g(n)$ is increasing, the proof can be done easily as in the case with (1.30). Otherwise, observe that $2^{\frac{1}{n^2}} \preceq 2^{\frac{1}{n}}$ but $\frac{1}{n^2} \not\preceq \frac{1}{n}$

Now consider (1.32). It is not true. As a counterexample, consider that $2^n \prec 2^{2n}$ but $n \not\prec 2n$. □

Problem 11. *Let a be a constant such that $a > 1$. Which of the following are true:*

$$\log_a \phi(n) \approx \log_a \psi(n) \ \Rightarrow\ \phi(n) \approx \psi(n) \tag{1.33}$$
$$\log_a \phi(n) \preceq \log_a \psi(n) \ \Rightarrow\ \phi(n) \preceq \psi(n) \tag{1.34}$$
$$\log_a \phi(n) \prec \log_a \psi(n) \ \Rightarrow\ \phi(n) \prec \psi(n) \tag{1.35}$$
$$\phi(n) \approx \psi(n) \ \Rightarrow\ \log_a \phi(n) \approx \log_a \psi(n) \tag{1.36}$$
$$\phi(n) \preceq \psi(n) \ \Rightarrow\ \log_a \phi(n) \preceq \log_a \psi(n) \tag{1.37}$$
$$\phi(n) \prec \psi(n) \ \Rightarrow\ \log_a \phi(n) \prec \log_a \psi(n) \tag{1.38}$$

for all asymptotically positive functions $\phi(n)$ and $\psi(n)$.

Solution:
Let $\phi(n) = a^{f(n)}$ and $\psi(n) = a^{g(n)}$, which means that $\log_a \phi(n) = f(n)$ and $\log_a \psi(n) = g(n)$. Consider (1.22) and conclude that (1.33) is not true. Consider (1.30) and conclude that (1.36) is true if $\psi(n)$ is increasing, and false otherwise. Consider (1.23) and conlude that (1.34) is not true. Consider (1.31) and conclude that (1.37) is true if $\psi(n)$ is increasing, and false otherwise. Consider (1.24) and conclude that (1.35) is true if $\psi(n)$ is increasing, and false otherwise. Consider (1.32) and conlude that (1.38) is not true. □

Problem 12. *Prove that for any two asymptotically positive functions $f(n)$ and $g(n)$, $f(n) \approx g(n)$ iff $f(n) \preceq g(n)$ and $f(n) \succeq n$.*

Solution:
In one direction, assume that $f(n) \approx g(n)$. Then there exist positive constants c_1 and c_2 and some n_0, such that:

$$0 \leq c_1.g(n) \leq f(n) \leq c_2.g(n),\ \forall n \geq n_0$$

It follows that,

$$0 \leq c_1.g(n) \leq f(n),\ \forall n \geq n_0 \tag{1.39}$$
$$0 \leq f(n) \leq c_2.g(n),\ \forall n \geq n_0 \tag{1.40}$$

In the other direction, assume that $f(n) \preceq g(n)$ and $f(n) \succeq g(n)$. Then there exists a positive constant c' and some n_0', such that:

$$0 \leq f(n) \leq c'.g(n), \forall n \geq n_0'$$

and there exists a positive constant c'' and some n_0'', such that:

$$0 \leq c''.g(n) \leq f(n), \forall n \geq n_0''$$

It follows that:

$$0 \leq c'.g(n) \leq f(n) \leq c''.g(n), \forall n \geq \max\{n_0', n_0''\}$$

□

Lemma 1 (Stirling's approximation)**.**

$$n! = \sqrt{2\pi n}\,\frac{n^n}{e^n}\left(1 + \Theta\left(\frac{1}{n}\right)\right) \tag{1.41}$$

□

Here, $\Theta\left(\frac{1}{n}\right)$ means any function that is in the set $\Theta\left(\frac{1}{n}\right)$.

Problem 13. *Prove that*

$$n! \approx n \lg n \tag{1.42}$$

Solution:
Use Stirling's approximation, ignoring the $\left(1 + \Theta\left(\frac{1}{n}\right)\right)$ factor, and take logarithm of both sides to obtain:

$$\lg(n!) = \lg(\sqrt{2\pi}) + \lg n + n \lg n - n \lg e$$

By Property 9 of the relations, $\lg(\sqrt{2\pi}) + \lg n + n \lg n - n \lg e \approx n \lg n$. □

Problem 14. *Prove that for any constant* $a > 1$*,*

$$a^n \prec n! \prec n^n \tag{1.43}$$

Solution:
Because of the factorial let us restrict n to positive integers.

$$\lim_{n\to\infty}\left(\frac{n.(n-1).(n-2)\ldots 2.1}{\underbrace{a\,.\,a\,.\,a\,\ldots\,a\,.\,a}_{n \text{ times}}}\right) = 0$$

$$\lim_{n\to\infty}\left(\frac{n.(n-1).(n-2)\ldots 2.1}{\underbrace{n\,.\,n\,.\,n\,\ldots\,n\,.\,n}_{n \text{ times}}}\right) = \infty$$

□

Problem 15 (polylogarithm versus constant power of n)**.** *Let* a, k *and* ϵ *be any constants, such that* $k \geq 1$*,* $a > 1$*, and* $\epsilon > 0$*. Prove that:*

$$(\log_a n)^k \prec n^\epsilon \tag{1.44}$$

Solution:
Take $\log_a$ of both sides. The left-hand side yields $k.\log_a \log_a n$ and the right-hand side yields $\epsilon.\log_a n$. But

$$k.\log_a \log_a n \prec \epsilon.\log_a n \tag{1.45}$$

because

$$\log_a \log_a n \prec \log_a n$$

Having in mind (1.35) we conclude immediately the desired relation holds. □

Problem 16 (polynomial versus exponent)**.** *Let* a *and* ϵ *be any constants, such that* $a > 1$ *and* $\epsilon > 0$*. Prove that:*

$$n^\epsilon \prec a^n \tag{1.46}$$

Solution:
Take $\log_a$ of both sides. The left-hand side yields $\epsilon.\log_a n$ and the right-hand side yields n. But

$$\epsilon.\log_a n \prec n \tag{1.47}$$

Having in mind (1.35) we conclude immediately the desired relation holds. □

Definition 1 (log-star function, [CLR00], pp. 36)**.** *Let the function* $\lg^{(i)} n$ *be defined recursively for nonnegative integers* i *as follows:*

$$\lg^{(i)} n = \begin{cases} n, & \text{if } i = 0 \\ \lg\left(\lg^{(i-1)} n\right), & \text{if } i > 0 \text{ and } \lg^{(i-1)} n > 0 \\ \text{undefined}, & \text{if } i > 0 \text{ and } \lg^{(i-1)} n < 0 \text{ or } \lg^{(i-1)} n \text{ is undefined} \end{cases}$$

Then

$$\lg^* n = \min\left\{ i \geq 0 \mid \lg^{(i)} n \leq 1 \right\}$$

□

According to this definition,

$$\begin{aligned}
\lg^* 2 &= 1, \text{ since } \lg^{(0)} 2 = 2 \text{ and } \lg^{(1)} 2 = \lg\left(\lg^{(0)} 2\right) = \lg(2) = 1 \\
\lg^* 3 &= 2, \text{ since } \lg^{(0)} 3 = 3 \text{ and } \lg\left(\lg^{(0)} 3\right) = \lg(\lg 3) = 0.6644\ldots \\
\lg^* 4 &= 2 \\
\lg^* 5 &= 3 \\
&\ldots \\
\lg^* 16 &= 3 \\
\lg^* 17 &= 4 \\
&\ldots \\
\lg^* 65536 &= 4 \\
\lg^* 65537 &= 5 \\
&\ldots \\
\lg^* 2^{65536} &= 5 \\
\lg^*\left(2^{65536}+1\right) &= 6 \\
&\ldots
\end{aligned}$$

Obviously, every real number t can be represented by a *tower of twos*:

$$t = 2^{2^{2^{\cdot^{\cdot^{\cdot^{2^s}}}}}}$$

where s is a real number such that $1 < s \leq 2$. The *height of the tower* is the number of elements in this sequence. For instance,

number	*its tower of twos*	*the height of the tower*
2	2	1
3	$2^{1.5849625007...}$	2
4	2^2	2
5	$2^{2^{1.2153232957...}}$	3
16	2^{2^2}	3
17	$2^{2^{2^{1.0223362884...}}}$	4
65536	$2^{2^{2^2}}$	4
65537	$2^{2^{2^{1.00000051642167...}}}$	5

Having that in mind, it is trivial to see that $\lg^* n$ is the height of the tower of twos of n.

Problem 17 ([CLR00], problem 2-3, pp. 38–39)**.** *Rank the following thirty functions by order of growth. That is, find the equivalence classes of the "$\approx$" relation and show their*

order by "$\succ$".

$\lg(\lg^* n)$	$2^{\lg^* n}$	$\left(\sqrt{2}\right)^{\lg n}$	n^2	$n!$	$(\lg n)!$
$\left(\frac{3}{2}\right)^n$	n^3	$\lg^2 n$	$\lg(n)!$	2^{2^n}	$n^{\frac{1}{\lg n}}$
$\ln\ln n$	$\lg^* n$	$n.2^n$	$n^{\lg\lg n}$	$\ln n$	1
$2^{\lg n}$	$(\lg n)^{\lg n}$	e^n	$4^{\lg n}$	$(n+1)!$	$\sqrt{\lg n}$
$\lg^*(\lg n)$	$2^{\sqrt{2\lg n}}$	n	2^n	$n\lg n$	$2^{2^{n+1}}$

Solution:

$2^{2^{n+1}} \succ 2^{2^n}$ because $2^{2^{n+1}} = 2^{2.2^n} = 2^{2^n} \times 2^{2^n}$.

$2^{2^n} \succ (n+1)!$ To see why, take logarithm to base two of both sides. The left-hand side becomes 2^n, the right-hand side becomes $\lg((n+1)!)$ By (1.41), $\lg((n+1)!) \approx (n+1)\lg(n+1)$, and clearly $(n+1)\lg(n+1) \approx n\lg n$. As $2^n \succ n\lg n$, by (1.35) we have $2^{2^n} \succ (n+1)!$

$(n+1)! \succ n!$ because $(n+1)! = (n+1) \times n!$

$n! \succ e^n$ by (1.43).

$e^n \succ n.2^n$. To see why, consider:

$$\lim_{n\to\infty} \frac{n.2^n}{e^n} = \lim_{n\to\infty} \frac{n}{\frac{e^n}{2^n}} = \lim_{n\to\infty} \frac{n}{\left(\frac{e}{2}\right)^n} = 0$$

$n.2^n \succ 2^n$

$2^n \succ \left(\frac{3}{2}\right)^n$. To see why, consider:

$$\lim_{n\to\infty} \frac{\left(\frac{3}{2}\right)^n}{2^n} = \lim_{n\to\infty} \left(\frac{3}{4}\right)^n = 0$$

$\left(\frac{3}{2}\right)^n \succ n^{\lg(\lg n)}$. To see why, take lg of both sides. The left-hand side becomes $n.\lg\left(\frac{3}{2}\right)$, the right-hand side becomes $\lg n.\lg(\lg n)$. Clearly, $\lg^2 n \succ \lg n.\lg(\lg n)$ and $n \succ \lg^2 n$ by (1.44). By transitivity, $n \succ \lg n.\lg(\lg n)$, and so $n.\lg\left(\frac{3}{2}\right) \succ \lg n.\lg(\lg n)$. Apply (1.35) and the desired conclusion follows.

$(\lg n)^{\lg n} = n^{\lg(\lg n)}$, which is obvious if we take lg of both sides. So, $(\lg n)^{\lg n} \approx n^{\lg(\lg n)}$.

$(\lg n)^{\lg n} \succ (\lg n)!$ To see why, substitute $\lg n$ with m, obtaining $m^m \succ m!$ and apply (1.43).

$(\lg n)! \succ n^3$. Take lg of both sides. The left-hand side becomes $\lg((\lg n)!)$. Substitute $\lg n$ with m, obtaining $\lg(m!)$. By (1.42), $\lg(m!) \approx m \lg m$, therefore $\lg((\lg n)!) \approx (\lg n).(\lg(\lg n))$. The right-hand side becomes $3.\lg n$. Compare $(\lg n).(\lg(\lg n))$ with $3.\lg n$:

$$\lim_{n\to\infty} \frac{3.\lg n}{(\lg n).(\lg(\lg n))} = \lim_{n\to\infty} \frac{3}{\lg(\lg n)} = 0$$

It follows that $(\lg n).(\lg(\lg n)) \succ 3.\lg n$. Apply (1.35) to draw the desired conclusion.

$n^3 \succ n^2$.

$n^2 \succ n \lg n$.

$\lg n! \approx n \lg n$ (see (1.42)).

$n \lg n \succ n$.

$n \approx 2^{\lg n}$ because $n = 2^{\lg n}$ by the properties of the logarithm.

$n \succ (\sqrt{2})^{\lg n}$ because $(\sqrt{2})^{\lg n} = 2^{\frac{1}{2}\lg n} = 2^{\lg\sqrt{n}} = \sqrt{n}$ and clearly $n \succ \sqrt{n}$.

$(\sqrt{2})^{\lg n} \succ 2^{\sqrt{2\lg n}}$. To see why, note that $\lg n \succ \sqrt{\lg n}$, therefore $\frac{1}{2}.\lg n \succ \sqrt{2}.\sqrt{\lg n} = \sqrt{2\lg n}$. Apply (1.24) and conclude that $2^{\frac{1}{2}.\lg n} \succ 2^{\sqrt{2\lg n}}$, *i.e.* $(\sqrt{2})^{\lg n} \succ 2^{\sqrt{2\lg n}}$.

$2^{\sqrt{2\lg n}} \succ \lg^2 n$. To see why, take lg of both sides. The left-hand side becomes $\sqrt{2\lg n}$ and the right-hand side becomes $\lg(\lg^2 n) = 2.\lg(\lg n)$. Substitute $\lg n$ with m: the left-hand side becomes $\sqrt{2m} = \sqrt{2}\sqrt{m} = \sqrt{2}.m^{\frac{1}{2}}$ and the right-hand side becomes $2\lg m$. By (1.44) we know that $m^{\frac{1}{2}} \succ \lg m$, therefore $\sqrt{2}.m^{\frac{1}{2}} \succ 2\lg m$, therefore $\sqrt{2m} \succ 2\lg m$, therefore $\sqrt{2\lg n} \succ \lg(\lg^2 n)$. Having in mind (1.35) we draw the desired conclusion.

$\lg^2 n \succ \ln n$. To see this is true, observe that $\ln n = \frac{\lg n}{\lg e}$.

$\ln n \succ \sqrt{\lg n}$.

$\sqrt{\lg n} \succ \ln\ln n$. The left-hand side is $\sqrt{\frac{\ln n}{\ln 2}}$. Substitute $\ln n$ with m and the claim becomes $\frac{1}{\sqrt{\ln 2}}.\sqrt{m} \succ \ln m$, which follows from (1.44).

$\ln\ln n \succ 2^{\lg^* n}$. To see why this is true, note that $\ln\ln n \approx \lg\lg n$ and rewrite the claim as $\lg\lg n \succ 2^{\lg^* n}$. Take lg of both sides. The left-hand side becomes $\lg\lg\lg n$, *i.e.* a triple logarithm. The right-hand side becomes $\lg^* n$. If we think of n as a tower of twos, it is obvious that the triple logarithm decreases the height of the tower with three, while, as we said before, the log-star measures the height of the tower. Clearly, the latter is *much* smaller than the former.

$2^{\lg^* n} \succ \lg^* n$. Clearly, for any increasing function $f(n)$, $2^{f(n)} \succ f(n)$.

$\lg^* n \approx \lg^*(\lg n)$. Think of n as a tower of twos and note that the difference in the height of n and $\lg n$ is one. Therefore, $\lg^*(\lg n) = (\lg^* n) - 1$.

$\lg^* n \succ \lg(\lg^* n)$. Substitute $\lg^* n$ with $f(n)$ and the claim becomes $f(n) \succ \lg f(n)$ which is clearly true since $f(n)$ is increasing.

$\lg(\lg^* n) \succ 1$.

$1 \approx n^{\frac{1}{\lg n}}$. Note that $n^{\frac{1}{\lg n}} = 2$: take lg of both sides, the left-hand side becomes $\lg\left(n^{\frac{1}{\lg n}}\right) = \frac{1}{\lg n}.\lg n = 1$ and the right-hand side becomes $\lg 2 = 1$. □

Problem 18. *Give an example of a function* $f(n)$, $n \in \mathbb{N}^+$, *such that for function* $g(n)$ *among the thirty functions from Problem 17,* $f(n) \not\preceq g(n)$ *and* $f(n) \not\succeq g(n)$.

Solution:
For instance,

$$f(n) = \begin{cases} 2^{2^{n+2}}, & \text{if } n \text{ is even} \\ \frac{1}{n}, & \text{if } n \text{ is odd} \end{cases}$$

□

Problem 19. *Is it true that for any asymptotically positive functions* $f(n)$ *and* $g(n)$, $f(n) + g(n) \approx \min(f(n), g(n))$*?*

Solution:
No. As a counterexample, consider $f(n) = n$ and $g(n) = 1$. Then $\min(f(n), g(n)) = 1$, $f(n) + g(n) = n + 1$, and certainly $n + 1 \not\approx 1$. □

Problem 20. *Is it true that for any asymptotically positive function* $f(n)$, $f(n) \preceq (f(n))^2$*?*

Solution:
If $f(n)$ is increasing, it is trivially true. If it is decreasing, however, it may not be true: consider (1.17). □

Problem 21. *Is it true that for any asymptotically positive function* $f(n)$, $f(n) \approx f(\frac{n}{2})$*?*

Solution:
No. As a counterexample, consider $f(n) = 2^n$. Then $f(\frac{n}{2}) = 2^{\frac{n}{2}}$. As we already saw, $2^n \not\approx 2^{\frac{n}{2}}$. □

Problem 22. *Compare the growth of* $n^{\lg n}$ *and* $(\lg n)^n$.

Solution:
Take logarithm of both sides. The left-hand side becomes $(\lg n)(\lg n) = \lg^2 n$, the right-hand side, $n.\lg(\lg n)$. As $n.\lg(\lg n) \succ \lg^2 n$, it follows that $(\lg n)^n \succ n^{\lg n}$. □

Problem 23. *Compare the growth of* $n^{\lg\lg\lg n}$ *and* $(\lg n)!$

Solution:
Take lg of both sides. The left-hand side becomes $(\lg n).(\lg\lg\lg n)$, the right-hand side becomes $\lg((\lg n)!)$. Substitute $\lg n$ with m is the latter expression to get $\lg((m)!) \approx m \lg m$. And that is $(\lg n).(\lg\lg n)$. Since $(\lg n).(\lg\lg n) \succ (\lg n).(\lg\lg\lg n)$, it follows that $(\lg n)! \succ n^{\lg\lg\lg n}$. □

Problem 24. *Let* $n!! = (n!)!$. *Compare the growth of* $n!!$ *and* $(n-1)!! \times ((n-1)!)^{n!}$.

Solution:
Let $(n-1)! = \nu$. Then $n! = n\nu$. We compare

$$n!! \quad \text{vs} \quad (n-1)!! \times ((n-1)!)^{n!}$$

$$(n\nu)! \quad \text{vs} \quad \nu! \times \nu^{n\nu}$$

Apply Stirling's approximation to both sides to get:

$$\sqrt{2\pi n\nu}\,\frac{(n\nu)^{n\nu}}{e^{n\nu}} \quad \text{vs} \quad \sqrt{2\pi\nu}\,\frac{\nu^{\nu}}{e^{\nu}} \times \nu^{n\nu}$$

$$\sqrt{2\pi n\nu}\,(n\nu)^{n\nu} \quad \text{vs} \quad \sqrt{2\pi\nu}\,e^{(n-1)\nu} \times \nu^{\nu} \times \nu^{n\nu}$$

Divide by $\sqrt{2\pi\nu}\,\nu^{n\nu}$ both sides:

$$\sqrt{n}\,n^{n\nu} \quad \text{vs} \quad e^{(n-1)\nu} \times \nu^{\nu}$$

Ignore the $\sqrt{n}$ factor on the left. If we derive without it that the left side grows faster, surely it grows even faster with it. So, consider:

$$n^{n\nu} \quad \text{vs} \quad e^{(n-1)\nu} \times \nu^{\nu}$$

Raise both sides to $\frac{1}{\nu}$:

$$n^{n} \quad \text{vs} \quad e^{n-1} \times \nu$$

That is,

$$n^{n} \quad \text{vs} \quad e^{n-1} \times (n-1)!$$

Apply Stirling's aproximation second time to get:

$$n^{n} \quad \text{vs} \quad e^{n-1} \times \sqrt{2\pi(n-1)}\,\frac{(n-1)^{n-1}}{e^{n-1}}$$

That is,

$$n^{n} \quad \text{vs} \quad \sqrt{2\pi(n-1)}\,(n-1)^{n-1}$$

Since $\sqrt{2\pi(n-1)}\,(n-1)^{n-1} \approx (n-1)^{\left(n-\frac{1}{2}\right)}$, we have

$$n^{n} \quad \text{vs} \quad (n-1)^{\left(n-\frac{1}{2}\right)}$$

Clearly, $n^{n} \succ (n-1)^{\left(n-\frac{1}{2}\right)}$, therefore $n!! \succ (n-1)!! \times ((n-1)!)^{n!}$. □

Lemma 2. *The function series:*

$$S(x) = \frac{\ln x}{x} + \frac{\ln^2 x}{x^2} + \frac{\ln^3 x}{x^3} + \dots$$

is convergent for $x > 1$. *Furthermore,* $\lim_{x\to\infty} S(x) = 0$.

Proof:
It is well known that the series

$$S'(x) = \frac{1}{x} + \frac{1}{x^2} + \frac{1}{x^3} + \dots$$

called *geometric series* is convergent for $x > 1$ and $S'(x) = \frac{1}{x-1}$ when $x > 1$. Clearly, $\lim_{x\to\infty} S'(x) = 0$. Consider the series

$$S''(x) = \frac{1}{\sqrt{x}} + \frac{1}{(\sqrt{x})^2} + \frac{1}{(\sqrt{x})^3} + \dots \tag{1.48}$$

It is a geometric series and is convergent for $\sqrt{x} > 1$, *i.e.* $x > 1$, and $\lim_{x\to\infty} S''(x) = 0$. Let us rewrite $S(x)$ as

$$S(x) = \frac{1}{\sqrt{x}.\frac{\sqrt{x}}{\ln x}} + \frac{1}{(\sqrt{x})^2.\left(\frac{\sqrt{x}}{\ln x}\right)^2} + \frac{1}{(\sqrt{x})^3.\left(\frac{\sqrt{x}}{\ln x}\right)^3} + \dots \tag{1.49}$$

For each term $f_k(x) = \frac{1}{(\sqrt{x})^k.\left(\frac{\sqrt{x}}{\ln x}\right)^k}$ of $S(x)$, $k \geq 1$, for large enough x, it is the case that $f_k(x) < g_k(x)$ where $g_k(x) = \frac{1}{\left(\sqrt{x}\right)^k}$ is the k^{th} term of $S''(x)$. To see why this is true, consider (1.44). Then the fact that $S''(x)$ is convergent and $\lim_{x\to\infty} S''(x) = 0$ implies the desired conclusion. □

Problem 25 ([Knu73], pp. 107)**.** *Prove that* $\sqrt[n]{n} \approx 1$.

Solution:
We will show an even stronger statement: $\lim_{n\to\infty} \sqrt[n]{n} = 1$. It is known that:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

Note that $\sqrt[n]{n} = e^{\ln \sqrt[n]{n}} = e^{\left(\frac{\ln n}{n}\right)}$.

$$e^{\left(\frac{\ln n}{n}\right)} = 1 + \underbrace{\frac{\ln n}{n} + \frac{\left(\frac{\ln n}{n}\right)^2}{2!} + \frac{\left(\frac{\ln n}{n}\right)^3}{3!} + \dots}_{T(n)}$$

Lemma 2 implies $\lim_{n\to\infty} T(n) = 0$. It follows that $\lim_{n\to\infty} \sqrt[n]{n} = 1$. □
We can also say that $\sqrt[n]{n} = 1 + O\left(\frac{\lg n}{n}\right)$, $\sqrt[n]{n} = 1 + \frac{\lg n}{n} + O\left(\frac{\lg^2 n}{n^2}\right)$, *etc*, where the big-Oh notation stands for any function of the set.

Problem 26 ([Knu73], pp. 107)**.** *Prove that* $n\left(\sqrt[n]{n}-1\right) \approx \ln n$.

Solution:
As

$$\sqrt[n]{n} = 1 + \frac{\ln n}{n} + \frac{\left(\frac{\ln n}{n}\right)^2}{2!} + \frac{\left(\frac{\ln n}{n}\right)^3}{3!} + \dots$$

it is the case that:

$$\sqrt[n]{n} - 1 = \frac{\ln n}{n} + \frac{\left(\frac{\ln n}{n}\right)^2}{2!} + \frac{\left(\frac{\ln n}{n}\right)^3}{3!} + \dots$$

Multiply by n to get:

$$n\left(\sqrt[n]{n}-1\right) = \ln n + \underbrace{\frac{(\ln n)^2}{2!n} + \frac{(\ln n)^3}{3!n^2} + \dots}_{T(n)}$$

Note that $\lim_{n\to\infty} T(n) = 0$ by an obvious generalisation of Lemma 2. The claim follows immediately. □

Problem 27. *Compare the growth of* n^n, $(n+1)^n$, n^{n+1}, *and* $(n+1)^{n+1}$.

Solution:
$n^n \approx (n+1)^n$ because

$$\lim_{n\to\infty} \frac{(n+1)^n}{n^n} = \lim_{n\to\infty} \left(\frac{n+1}{n}\right)^n = \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n = e$$

Clearly, $n^n \prec n^{(n+1)} = n.n^n$. And $n^{(n+1)} \approx (n+1)^{(n+1)}$:

$$\lim_{n\to\infty} \frac{(n+1)^{n+1}}{n^{n+1}} = \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^{n+1} = \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n \lim_{n\to\infty} \left(1 + \frac{1}{n}\right) = e.1 = e$$

□

Problem 28. *Let* k *be a positive integer constant. Prove that*

$$1 + k + k^2 + k^3 + \dots + k^n = \Theta(k^n)$$

Solution:
First assume n is an integer variable. Then

$$1 + k + k^2 + k^3 + \dots + k^n = \frac{k^{n+1} - 1}{k - 1} = \Theta(k^n)$$

The result can obviously be extended for real n, provided we define appropriately the sum. For instance, let the sum be

$$S(n) = 1 + k + k^2 + k^3 + \dots + k^{\lfloor n-2 \rfloor} + k^{\lfloor n-1 \rfloor} + k^n$$

By the above result, $S(n) = k^n + \Theta\left(k^{\lfloor n-1 \rfloor}\right) = \Theta(k^n)$. □

Chapter 2

Iterative Algorithms

In this section we compute the asymptotic running time of algorithms that use the "for" and "while" branching statements but make no calls to other algorithms or themselves. Consider the following trivial algorithm.

Algorithm ADD-1(n: nonnegative integer)
```
1  a ← 0
2  for i ← 1 to n
3      a ← a + i
4  return a
```

We assume the expression on line 3 requires constant time to be executed regardless of how large n is. Let $c = \text{const}$ be that time. We further assume that line 2 does not take *any* time – certainly, not a realistic assumption since at every iteration the control variable i must be incremented and then compared against n but we nevertheless make it. Under these two assumptions the total execution time of the **for** loop is precisely $c.n$. The running time of the algorithm is $c_1 + c.n$, where c_1 is the execution time of the assignment at line 1. Using the notatons from the previous section, the running time is $\Theta(n)$.

Now consider another algorithm:

Algorithm ADD-2(n: nonnegative integer)
```
1  return n * (n + 1)/2
```

Clearly, ADD-2 is equivalent to ADD-1 but the running time of ADD-2 is, under the said assumptions, constant. We denote constant running time by $\Theta(1)$[†]. It is not incorrect to say the running time of both algorithms is $O(n)$ but the big-Theta notation is superior as it grasps precisely—in the asymptotic sense—the algorithm's running time.

Consider the following iterative algorithm:

Algorithm ADD-3(n: nonnegative integer)
```
1  a ← 0
2  for i ← 1 to n
```

[†]All constants are bit-Theta of each other so we might have as well used $\Theta(1000)$ or $\Theta(0.0001)$ but we prefer the simplest form $\Theta(1)$.

```
3       for j ← 1 to n
4           a ← a + 1
5   return a
```

The execution time of lines 3–4 is $\Theta(n)$ for the same reason that Add-1 has running time $\Theta(n)$. Lines 3–4 are executed n times because the first control variable, *viz.* i, takes n values $1, 2, \ldots, n$, and for each of those we have one execution of lines 3–4. Therefore, the running time of ADD-3 is $\Theta(n^2)$. Another way to derive that is to note that the running time of lines 2–4 is given by $\sum_{i=1}^{n}\sum_{j=1}^{n} c = c\sum_{i=1}^{n}\sum_{j=1}^{n} 1 = c.n^2 = \Theta(n^2)$.

Algorithm ADD-3 has two *nested cycles*. We can generalise that the running time of k nested cycles of the type

```
Algorithm ()
1   for i_1 ← 1 to n
2       for i_2 ← 1 to n
3           ...
4           for i_k ← 1 to n
5               expression
```

where **expression** is computed in $\Theta(1)$, has running time $\Theta(n^k)$.

Let us consider a modification of ADD-3:

```
Algorithm Add-4(n: nonnegative integer)
1   a ← 0
2   for i ← 1 to n
3       for j ← i to n
4           a ← a + 1
5   return a
```

The running time is determined by the number of times line 4 is executed, analogously to ADD-3:

$$\sum_{i=1}^{n}\sum_{j=i}^{n} 1 = \sum_{i=1}^{n}\Big(\underbrace{\sum_{j=1}^{n} 1}_{n} - \underbrace{\sum_{j=1}^{i-1} 1}_{i-1}\Big) = \sum_{i=1}^{n}(n-i+1) = \sum_{i=1}^{n}(n+1) - \sum_{i=1}^{n} i =$$

$$n(n+1) - \frac{n(n+1)}{2} = \frac{1}{2}n^2 + \frac{1}{2}n = \Theta(n^2)$$ (see Problem 4 on page 5.)

It follows that asymptotically ADD-4 has the same running time as ADD-3.

Consider the following algorithm:

```
Algorithm A1(n: nonnegative integer)
1   for i ← 1 to n
2       for j ← i + 1 to n
3           expression
```

where **expression** is computed in constant time c. The overall running time equals

$$\sum_{i=1}^{n}\sum_{j=i+1}^{n} c = c.\sum_{i=1}^{n}\Big(\underbrace{\sum_{j=1}^{n} 1}_{n} - \underbrace{\sum_{j=1}^{i} 1}_{i}\Big) = c.\sum_{i=1}^{n}(n-i) = c\left(\sum_{i=1}^{n} n - \sum_{i=1}^{n} i\right) =$$

$$c\left(n^2 - \frac{n(n+1)}{2}\right) = \frac{c}{2}n^2 - \frac{c}{2}n = \Theta(n^2)$$

If we are given an algorithm having input some positive integer n, a single set of nested cycles whose boundary conditions depend on n, and inside the nested cycles, constant time expression, a possible way of determining the running time is to change the expression to $a \leftarrow a + 1$, add an initial $a \leftarrow 0$ assignment, and determine the value of a at the end as a function of n. Typically that involves manipulation of sums. However, that method gives more detailed answer than necessary. Consider the following algorithm:

Algorithm A2(n: positive integer)

```
1  a ← 0
2  for i ← 1 to n − 1
3      for j ← i + 1 to n
4          for k ← 1 to j
5              a ← a + 1
6  return a
```

We are asked to determine a that A2 returns as a function of n. The answer clearly is $\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\sum_{k=1}^{j} 1$, we just need to find an equivalent closed form.

$$\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\sum_{k=1}^{j} 1 = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} j = \sum_{i=1}^{n-1}\left(\sum_{j=1}^{n} j - \sum_{j=1}^{i} j\right) =$$

$$\sum_{i=1}^{n-1}\left(\frac{1}{2}n(n+1) - \frac{1}{2}i(i+1)\right) =$$

$$\sum_{i=1}^{n-1}\left(\frac{1}{2}n(n+1)\right) - \frac{1}{2}\sum_{i=1}^{n-1}(i^2+i) =$$

$$\frac{1}{2}n(n+1)(n-1) - \frac{1}{2}\sum_{i=1}^{n-1} i^2 - \frac{1}{2}\sum_{i=1}^{n-1} i$$

But $\sum_{i=1}^{n} i^2 = \frac{1}{6}n(n+1)(2n+1)$, therefore $\sum_{i=1}^{n-1} i^2 = \frac{1}{6}(n-1)n(2n-1)$. Further, $\sum_{i=1}^{n-1} i =$

$\frac{1}{2}n(n-1)$, so we have

$$\frac{1}{2}n(n-1)(n+1) - \frac{1}{12}n(n-1)(2n-1) - \frac{1}{4}n(n-1) =$$
$$\frac{1}{2}n(n-1)\left(n+1-\frac{1}{6}(2n-1)-\frac{1}{2}\right) =$$
$$\frac{1}{12}n(n-1)(6n+3-2n+1) = \frac{1}{12}n(n-1)(4n+4) =$$
$$\frac{1}{3}n(n-1)(n+1)$$

That implies that the running time of A2 is $\Theta(n^3)$. Clearly A2 is equivalent to the following algorithm.

Algorithm A3(n: positive integer)
```
1  return  n(n-1)(n+1)/3
```

whose running time is $\Theta(1)$.

Algorithm A4(n: positive integer)
```
1  a ← 0
2  for i ← 1 to n
3      for j ← i+1 to n
4          for k ← i+j-1 to n
5              a ← a+1
6  return a
```

Problem 29. *Find the running time of algorithm* A4 *by determining the value of* a *it returns as a function of* n, $f(n)$. *Find a closed form for* $f(n)$.

Solution:

$$f(n) = \sum_{i=1}^{n}\sum_{j=i+1}^{n}\sum_{k=i+j-1}^{n} 1$$

Let us evaluate the innermost sum $\sum_{k=i+j-1}^{n} 1$. It is easy to see that the lower boundary $i+j-1$ may exceed the higher boundary n. If that is the case, the sum is zero because the index variable takes values from the empty set. More precisely, for any integer t,

$$\sum_{i=t}^{n} 1 = \begin{cases} n-t+1 & \text{, if } t \le n \\ 0 & \text{, else} \end{cases}$$

It follows that

$$\sum_{k=i+j-1}^{n} 1 = \begin{cases} n-i-j+2 & \text{, if } i+j-1 \le n \Leftrightarrow j \le n-i+1 \\ 0 & \text{, else} \end{cases}$$

Then

$$f(n) = \sum_{i=1}^{n} \sum_{j=i+1}^{n-i+1} (n+2-(i+j))$$

Now the innermost sum is zero when $i+1 > n-i+1 \Leftrightarrow 2i > n \Leftrightarrow i > \left\lfloor \frac{n}{2} \right\rfloor$, therefore the maximum i we have to consider is $\left\lfloor \frac{n}{2} \right\rfloor$:

$$f(n) = \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} \sum_{j=i+1}^{n-i+1} (n+2-(i+j)) =$$

$$(n+2) \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} \sum_{j=i+1}^{n-i+1} 1 - \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} i \left(\sum_{j=i+1}^{n-i+1} 1 \right) - \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} \sum_{j=i+1}^{n-i+1} j =$$

$$(n+2) \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} (n-i+1-(i+1)+1) - \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} i(n-i+1-(i+1)+1) -$$

$$\sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} \left(\sum_{j=1}^{n-i+1} j - \sum_{j=1}^{i} j \right) =$$

$$(n+2) \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} (n-2i+1) - \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} i(n-2i+1) -$$

$$\sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} \left(\frac{(n-i+1)(n-i+2)}{2} - \frac{i(i+1)}{2} \right) =$$

$$(n+2)(n+1) \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} 1 - 2(n+2) \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} i - (n+1) \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} i + 2 \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} i^2 -$$

$$\frac{1}{2} \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} \left((n+1)(n+2) - i(2n+3) + \cancel{i^2} - \cancel{i^2} - i \right) =$$

$$(n+2)(n+1) \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} 1 - (3n+5) \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} i + 2 \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} i^2 -$$

$$\frac{(n+1)(n+2)}{2} \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} 1 + \frac{(2n+4)}{2} \sum_{i=1}^{\left\lfloor \frac{n}{2} \right\rfloor} i =$$

$$\left\lfloor \frac{n}{2} \right\rfloor (n+1)(n+2) - (3n+5) \frac{\left\lfloor \frac{n}{2} \right\rfloor \left(\left\lfloor \frac{n}{2} \right\rfloor + 1 \right)}{2} + 2 \frac{\left\lfloor \frac{n}{2} \right\rfloor \left(\left\lfloor \frac{n}{2} \right\rfloor + 1 \right) \left(2 \left\lfloor \frac{n}{2} \right\rfloor + 1 \right)}{6} -$$

$$\frac{1}{2} \left\lfloor \frac{n}{2} \right\rfloor (n+1)(n+2) + (n+2) \frac{\left\lfloor \frac{n}{2} \right\rfloor \left(\left\lfloor \frac{n}{2} \right\rfloor + 1 \right)}{2} =$$

$$\frac{\left\lfloor \frac{n}{2} \right\rfloor (n+1)(n+2)}{2} - \frac{\left\lfloor \frac{n}{2} \right\rfloor \left(\left\lfloor \frac{n}{2} \right\rfloor + 1 \right) (2n+3)}{2} + \frac{\left\lfloor \frac{n}{2} \right\rfloor \left(\left\lfloor \frac{n}{2} \right\rfloor + 1 \right) \left(2 \left\lfloor \frac{n}{2} \right\rfloor + 1 \right)}{3}$$

When n is even, *i.e.* $n = 2k$ for some $k \in \mathbb{N}^+$, $\left\lfloor \frac{n}{2} \right\rfloor = k$ and so

$$f(n) = \frac{k(2k+1)(2k+2)}{2} - \frac{k(k+1)(4k+3)}{2} + \frac{k(k+1)(2k+1)}{3} =$$
$$\frac{k(k+1)(4k+2) - k(k+1)(4k+3)}{2} + \frac{k(k+1)(2k+1)}{3} =$$
$$k(k+1)\left(-\frac{1}{2} + \frac{2k+1}{3}\right) = \frac{k(k+1)(4k-1)}{6}$$

When n is odd, *i.e.* $n = 2k+1$ for some $k \in \mathbb{N}$, $\left\lfloor \frac{n}{2} \right\rfloor = k$ and so

$$f(n) = \frac{k(2k+2)(2k+3)}{2} - \frac{k(k+1)(4k+5)}{2} + \frac{k(k+1)(2k+1)}{3} =$$
$$\frac{k(k+1)(4k+6) - k(k+1)(4k+5)}{2} + \frac{k(k+1)(2k+1)}{3} =$$
$$k(k+1)\left(\frac{1}{2} + \frac{2k+1}{3}\right) = \frac{k(k+1)(4k+5)}{6}$$

Obviously, $f(n) = \Theta(n^3)$. □

Algorithm A5(n: positive integer)

```
1  a ← 0
2  for i ← 1 to n
3      for j ← i to n
4          for k ← n + i + j − 3 to n
5              a ← a + 1
6  return a
```

Problem 30. *Find the running time of algorithm* A4 *by determining the value of* a *it returns as a function of* n, $f(n)$. *Find a closed form for* $f(n)$.

Solution:
We have three nested **for** cycles and it is certainly true that $f(n) = O(n^3)$. However, now $f(n) \neq \Theta(n^3)$. It is easy to see that for any large enough n, line 5 is executed for only four values of the ordered triple $\langle i, j, k \rangle$. Namely,

$$\langle i, j, k \rangle \in \{\langle 1, 1, n-1 \rangle, \langle 1, 1, n \rangle, \langle 1, 2, n-1 \rangle, \langle 2, 1, n \rangle\}$$

because the condition in the innermost loop (line 5) requires that $i + j \leq 3$. So, $f(n) = 4$, thus $f(n) = \Theta(1)$. □

Problem 30 raises a question: does it make sense to compute the running time of an iterative algorithm by counting how many time the expression in the innermost loop is executed? At lines 2 and 3 of A5 there are condition evaluations and variable increments – can we

assume they take no time at all? Certainly, if that was a segment of a real-world program, the outermost two loops would be executed $\Theta(n^2)$ times, unless some sort of optimisation was applied by the compiler. Anyway, we *postulate* that the running time is evaluated by counting how many times the innermost loop is executed. Whether that is a realistic model for real-world computation or not, is a side issue.

Algorithm A6($a_1, a_2, \ldots a_n$: array of positive distinct integers, $n \geq 3$)

```
1  S: a stack of positive integers
2  (* P(S) is a predicate. P(S) is true iff there are at least two *)
3  (* elements in S and top(S) > next-to-top(S) *)
4  push(a_1, S)
5  push(a_2, S)
6  for i ← 3 to n
7      while P(S) do
8          pop(S)
9      push(a_i, S)
```

Problem 31. *Find the asymptotic growth rate of running time of* A6. *Assume the predicate* P *is evaluated in* $\Theta(1)$ *time and the push and pop operations are executed in* $\Theta(1)$ *time.*

Solution:
Certainly, the running time is $O(n^2)$ because the outer loop runs $\Theta(n)$ times and the inner loop runs in $O(n)$ time: note that for each concrete i, the inner loop (line 8) cannot be executed more than $n-2$ times sinse there are at most n elements in S and each execution of line 8 removes one element from S.

However, a more precise analysis is possible. Observe that each element of the array is being pushed in S and *may be* popped out of S later but only once. It follows that line 8 cannot be exesuted more than n times altogether, *i.e.* for all i, and so the algorithm runs in $\Theta(n)$ time. □

Algorithm A7($a_1, a_2, \ldots a_n$: array of positive distinct integers, x: positive integer)

```
1  i ← 1
2  j ← n
3  while i ≤ j do
4      k ← ⌊(i+j)/2⌋
5      if x = a_k
6          return k
7      else if x < a_k
8          j ← k − 1
9      else i ← k + 1
10 return −1
```

Problem 32. *Find the asymptotic growth rate of running time of* A7.

Solution:
The following claim is a loop invariant for A7:

> *For every iteration of the* ***while*** *loop of* A7*, if the iteration number is* t*,* $t \geq 0$*, it is the case that:*
>
> $$j - i < \frac{n}{2^t} \tag{2.1}$$

We prove it by induction on t. The basis is $t = 0$, *i.e.* the fisrt time the execution reaches line 3. Then j is n, i is 1, and indeed $n - 1 < \frac{n}{2^0} = n$. Assume that at iteration t, $t \geq 1$, (2.1) holds. Consider iteration $t+1$. There are two ways to get from iteration t to iteration $t+1$ and we consider them in separate cases.

Case I: we exit iteration t through line 8 In this case, j becomes $\left\lfloor \frac{i+j}{2} \right\rfloor - 1$ and i stays the same when going from iteration t to iteration $t+1$.

$$\begin{aligned}
\frac{j-i}{2} &< \frac{n}{2^{t+1}} && \text{directly from (2.1)} \\
\frac{j+i-2i}{2} &< \frac{n}{2^{t+1}} \\
\frac{j+i}{2} - i &< \frac{n}{2^{t+1}} \\
\underbrace{\left\lfloor \frac{j+i}{2} \right\rfloor - 1}_{\text{the new } j} - i &< \frac{n}{2^{t+1}} && \text{since } \lfloor m \rfloor - 1 \leq m,\ \forall m \in \mathbb{R}^+
\end{aligned}$$

And so the induction step follows from the induction hypothesis.

Case II: we exit iteration t through line 9 In this case, j stays the same and i becomes $\left\lfloor \frac{i+j}{2} \right\rfloor + 1$ when going from iteration t to iteration $t+1$.

$$\begin{aligned}
\frac{j-i}{2} &< \frac{n}{2^{t+1}} && \text{directly from (2.1)} \\
\frac{2j-j-i}{2} &< \frac{n}{2^{t+1}} \\
j - \frac{j+i}{2} &< \frac{n}{2^{t+1}} \\
j - \underbrace{\left(\left\lfloor \frac{j+i}{2} \right\rfloor + 1 \right)}_{\text{the new } i} &< \frac{n}{2^{t+1}} && \text{since } \lfloor m \rfloor + 1 \geq m,\ \forall m \in \mathbb{R}^+
\end{aligned}$$

And so the induction step follows from the induction hypothesis.

Having proven (2.1), we see that $2^t < \frac{n}{j-i}$. It is obvious that $j - i \geq 1$ at the beginning of any iteration of the loop, so $\frac{n}{j-i} \leq n$, and therefore $2^t < n \Leftrightarrow t < \lceil \lg n \rceil$. Recall that t is, after A7 finishes, the number of times the loop has been executed. It follows that the running time of A7 is $O(\lg n)$. The logarithmic bound is not tight in general – obviously,

the best-case running time is $\Theta(1)$. The worst-case running time, however, is $\Omega(\lg n)$, so the worst case running time is $\Theta(\lg n)$. Now we prove the worst-case running time is $\Omega(n)$.

The following claim is a loop invariant for A7:

> *For every iteration of the* ***while*** *loop of* A7*, if the iteration number is* t*,* $t \geq 0$*, it is the case that:*
>
> $$\frac{n}{2^{t+1}} - 4 < j - i \tag{2.2}$$

We prove it by induction on t. The basis is $t = 0$, *i.e.* the fisrt time the execution reaches line 3. Then j is n, i is 1, and indeed $\frac{n}{2^{1+0}} = \frac{n}{2} < n - 1$, for large enough n. Assume that at iteration t, $t \geq 1$, (2.2) holds. Consider iteration $t + 1$. There are two ways to get from iteration t to iteration $t + 1$ and we consider them in separate cases.

Case I: we exit iteration t through line 8 In this case, j becomes $\left\lfloor \frac{i+j}{2} \right\rfloor - 1$ and i stays the same when going from iteration t to iteration $t + 1$.

$$\begin{aligned}
&\frac{n}{2^{t+2}} - 2 < \frac{j-i}{2} \quad \text{directly from (2.2)} \\
&\frac{n}{2^{t+2}} - 2 < \frac{j+i-2i}{2} \\
&\frac{n}{2^{t+2}} - 2 < \frac{j+i}{2} - i \\
&\frac{n}{2^{t+2}} - 4 < \frac{j+i}{2} - 2 - i \\
&\frac{n}{2^{t+2}} - 4 < \underbrace{\left\lfloor \frac{j+i}{2} \right\rfloor - 1}_{\text{the new } j} - i \quad \text{since } m - 2 \leq \lfloor m \rfloor - 1,\ \forall m \in \mathbb{R}^+
\end{aligned}$$

Case II: we exit iteration t through line 9 In this case, j stays the same and i becomes $\left\lfloor \frac{i+j}{2} \right\rfloor + 1$ when going from iteration t to iteration $t + 1$.

$$\begin{aligned}
&\frac{n}{2^{t+2}} - 2 < \frac{j-i}{2} \\
&\frac{n}{2^{t+2}} - 2 < \frac{2j-j-i}{2} \\
&\frac{n}{2^{t+2}} - 2 < j - \frac{j+i}{2} \\
&\frac{n}{2^{t+2}} - 4 < j - \frac{j+i}{2} - 2 \\
&\frac{n}{2^{t+2}} - 4 < j - \left(\frac{j+i}{2} + 2 \right) \\
&\frac{n}{2^{t+2}} - 4 < j - \underbrace{\left(\left\lfloor \frac{j+i}{2} \right\rfloor + 1 \right)}_{\text{the new } i} \quad \text{since } m + 2 \geq \lfloor m \rfloor + 1,\ \forall m \in \mathbb{R}^+
\end{aligned}$$

Having proven (2.2), it is trivial to prove that in the worst case, *e.g.* when x is not in the array, the loop is executed $\Omega(\lg n)$ times. □

Problem 33. *Determine the asymptotic running time of the following programming segment:*

```
s = 0;
for(i = 1; i * i <= n; i ++)
  for(j = 1; j <= i; j ++)
   s += n + i - j;
return s;
```

Solution:
The segment is equivalent to:

```
s = 0;
for(i = 1; i <= floor(sqrt(n)); i ++)
  for(j = 1; j <= i; j ++)
   s += n + i - j;
return s;
```

As we already saw, the running time is $\Theta\left(\left(\sqrt{n}\right)^2\right)$ and that is $\Theta(n)$. □

Problem 34. *Assume that* $A_{n\times n}$, $B_{n\times n}$, *and* $C_{n\times n}$ *are matrices of integers. Determine the asymptotic running time of the following programming segment:*

```
for(i = 1; i <= n; i ++)
 for(j = 1; j <= n; j ++) {
  s = 0;
  for(k = 1; k <= n; k ++)
    s += A[i][k] * B[k][j];
  C[i][j] = s; }
return s;
```

Solution:
Having in mind the analysis of ADD-3 on page 20, clearly this is a $\Theta(n^3)$ algorithm. However, if consider the order of growth as a function of the *length of the input*, the order of growth is $\Theta\left(m^{\frac{3}{2}}\right)$, where m is the length of the input, *i.e.* m is the order of the number of elements in the matrices and $m = \Theta(n^2)$. □

Algorithm A8($a_1, a_2, \ldots a_n$: array of positive integers)

```
1  s ← 0
2  for i ← 1 to n − 4
3      for j ← i to i + 4
4          for k ← i to j
5              s ← s + a_i
```

Problem 35. *Determine the running time of algorithm* A8.

Solution:
The outermost loop is executed $n-4$ times (assume large enough n). The middle loop is executed 5 times precisely. The innermost loop is executed 1, 2, 3, 4, or 5 times for j equal to i, $i+1$, $i+2$, $i+3$, and $i+4$, respectively. Altogether, the running time is $\Theta(n)$. □

Algorithm A9(n: positive integer)

```
1  s ← 0
2  for i ← 1 to n − 4
3      for j ← 1 to i + 4
4          for k ← i to j
5              s ← s + 1
6  return s
```

Problem 36. *Determine the running time of algorithm* A9. *First determine the value it returns as a function of* n.

Solution:
We have to evaluate the sum:

$$\sum_{i=1}^{n-4}\sum_{j=1}^{i+4}\sum_{k=i}^{j}1$$

Having we mind that

$$\sum_{k=i}^{j}1=\begin{cases}j-i+1, & \text{if } j\geq i\\ 0, & \text{else}\end{cases}$$

we rewrite the sum as:

$$\sum_{i=1}^{n-4}\left(\underbrace{\sum_{j=1}^{i-1}\sum_{k=i}^{j}1}_{\text{this is }0}+\sum_{j=i}^{i+4}\sum_{k=i}^{j}1\right)=$$

$$\sum_{i=1}^{n-4}\sum_{j=i}^{i+4}(j-i+1)=$$

$$\sum_{i=1}^{n-4}\big((i-i+1)+(i+1-i+1)+(i+2-i+1)+(i+3-i+1)+(i+4-i+1)\big)=$$

$$\sum_{i=1}^{n-4}(1+2+3+4+5)=15(n-4)$$

Since the returned s is $15(n-4)$, the algorithm runs in $\Theta(n)$ time. □

Our notations from Chapter 1 can be generalised for two variables as follows. A bivariate function $f(n,m)$ is asymptotically positive iff

$$\exists n_0\exists m_0:\forall n\geq n_0\forall m\geq m_0, f(n,m)>0$$

Definition 2. *Let* $g(n, m)$ *be an asymptotically positive function with real domain and codomain. Then*

$$\Theta(g(n,m)) = \{f(n,m) \mid \exists c_1, c_2 > 0,\ \exists n_0, m_0 > 0 : \\ \forall n \geq n_0,\ 0 \leq c_1.g(n,m) \leq f(n,m) \leq c_2.g(n,m)\}$$

Pattern matching is a computational problem in which we are given a *text* and a *pattern* and we compute how many times or, in a more elaborate version, at what *shifts*, the pattern occurs in the text. More formally, we are given two arrays of characters T and P with lengths n and m, respectively, such that $n \geq m$. For any k, $1 \leq k \leq n - m + 1$, we have a shift at position k iff:

$$\begin{aligned} T[k] &= P[k] \\ T[k+1] &= P[k+1] \\ &\dots \\ T[k+m-1] &= P[k+m-1] \end{aligned}$$

The problem then is to determine all the valid shifts. Consider the following algorithm for that problem.

Algorithm NAIVE-PATTERN-MATHING($T[1..n]$: characters, $P[1..m]$: characters)

```
1  (* assume n ≥ m *)
2  for i ← 1 to n − m + 1
3      if T[i, i + 1, ..., i + m − 1] = P
4          print "shift at" i
```

Problem 37. *Determine the running time of algorithm* NAIVE-PATTERN-MATHING.

Solution:
The algorithm is ostensibly $\Theta(n)$ because it has a single loop with the loop control variable running from 1 to n. That analysis, however, is wrong because the comparison at line 3 cannot be performed in constant time. Have in mind that m can be as long as n. Therefore, the algorihm is in fact:

Algorithm NAIVE-PATTERN-MATHING-1($T[1..n]$: characters, $P[1..m]$: characters)

```
1  (* assume n ≥ m *)
2  for i ← 1 to n − m + 1
3      Match ← TRUE
4      for j ← 1 to m
5          if T[i + j − 1] ≠ P[j]
6              Match ← FALSE
7      if Match
8          print "shift at" i
```

For obvious reasons this is a $\Theta((n-m).m)$ algorithm: both the best-case and the worst-case running times are $\Theta((n-m).m)$[†]. Suppose we improve it to:

Algorithm NAIVE-PATTERN-MATHING-2(T[1..n]: characters, P[1..m]: characters)

```
1   (* assume n ≥ m *)
2   for i ← 1 to n − m + 1
3       Match ← TRUE
4       j ← 1
5       while Match AND j ≤ m do
6           if T[i + j − 1] = P[j]
7               j ← j + 1
8           else
9               Match ← FALSE
10      if Match
11          print "shift at" i
```

NAIVE-PATTERN-MATHING-2 has the advantage that once a mismatch is found (line 9) the inner loop "breaks". Thus the best-case running time is $\Theta(n)$. A best case, for instance, is:

$$T = \underbrace{a\,a\,\ldots\,a}_{n \text{ times}} \quad \text{and} \quad P = \underbrace{b\,b\,\ldots\,b}_{m \text{ times}}$$

However, the worst case running time is still $\Theta((n-m).m)$. A worst case is, for instance:

$$T = \underbrace{a\,a\,\ldots\,a}_{n \text{ times}} \quad \text{and} \quad P = \underbrace{a\,a\,\ldots\,a}_{m \text{ times}}$$

It is easy to prove that $(n-m).m$ is maximised when m varies and n is fixed for $m \approx \frac{n}{2}$ and achieves maximum value $\Theta(n^2)$. It follows that all the naive string matchings are, at worst, quadratic algorithms. □

It is known that faster algorithms exist for the pattern matching problem. For instance, the Knuth-Morris-Pratt [KMP77] algorithm that runs in $\Theta(n)$ in the worst case.

Problem 38. *For any two strings* x *and* y *of the same length, we say that* x is a circular shift of y *iff* y *can be broken into substrings—one of them possibly empty—*y_1 *and* y_2*:*

$$y = y_1\,y_2$$

such that $x = y_2\,y_1$*. Find a linear time algorithm,* i.e. $\Theta(n)$ *in the worst case, that computes whether* x *is a circular shift of* y *or not. Assume that* $x \neq y$*.*

Solution:
Run the linear time algorithm for string matching of Knuth-Morris-Pratt with input $y\,y$ (y concatenated with itself) as text and x as pattern. The algorithm will output one or more valid shifts iff x is a circular shift of y, and zero valid shifts, otherwise. To see why, consider

[†]Algorithms that have the same—in asymptotic terms—running time for all inputs of the same length are called *oblivious*.

the concatenation of y with itself when it is a circular shift of x for some y_1 and y_2, such that $y = y_1 y_2$ and $x = y_2 y_1$:

$$y \quad y \quad = \quad y_1 \quad \underbrace{y_2 \quad y_1}_{\text{this is } x} \quad y_2$$

The running time is $\Theta(2n)$, *i.e.* $\Theta(n)$, at worst. □

Chapter 3

Recursive Algorithms and Recurrence Relations

3.1 Preliminaries

A *recursive algorithm* is an algorithm that calls itself, one or more times on smaller inputs. To prevent an infinite chain of such calls there has to be a value of the input for which the algorithm does not call itself.

A *recurrence relation in one variable* is an equation, *i.e.* there is an "=" sign "in the middle", in which a function of the variable is equated to an expression that includes the same function on smaller value of the variable. In addition to that for some basic value of the variable, typically one or zero, an explicit value for the function is defined – that is the initial condition[†]. The variable is considered by default to take nonnegative integer values, although one can think of perfectly valid recurrence relations in which the variable is real.

Typically, in the part of the relation that is not the initial condition, the function of the variable is written on the left-hand side of the "=" sign as, say, $T(n)$, and the expression, on the right-hand side, *e.g.* $T(n) = T(n-1) + 1$. If the initial condition is, say, $T(0) = 0$, we typically write:

$$\begin{aligned} &T(n) = T(n-1) + 1,\ \forall n \in \mathbb{N}^+ \\ &T(0) = 0 \end{aligned} \tag{3.1}$$

It is not formally incorrect to write the same thing as:

$$\begin{aligned} &T(n-1) = T(n-2) + 1,\ \forall n \in \mathbb{N}^+, n \neq 1 \\ &\quad T(0) = 0 \end{aligned}$$

The equal sign is interpreted as an assignment from right to left, just as the equal sign in the C programming language, so the following "unorthodox" way of describing the same

[†]Note there can be more than one initial condition as in the case with the famous Fibonacci numbers:

$$\begin{aligned} &F(n) = F(n-1) + F(n-2),\ \forall n \in \mathbb{N}^+, n \neq 1 \\ &F(1) = 1 \\ &F(0) = 0 \end{aligned}$$

The number of initial conditions is such that the initial conditions prevent "infinite descent".

relation is *discouraged*:

$$T(n-1)+1 = T(n),\ \forall n \in \mathbb{N}^+$$
$$0 = T(0)$$

Each recurrence relation defines an infinite numerical sequence, provided the variable is integer. For example, (3.1) defines the sequence $0, 1, 2, 3, \ldots$. Each term of the relation, except for the terms defined by the initial conditions, is defined recursively, *i.e.* in terms of smaller terms, hence the name. To *solve* a recurrence relation means to find a non-recursive expression for the same function – one that defines the same sequence. For example, the solution of (3.1) is $T(n) = n$.

It is natural to describe the running time of a recursive algorithm by some recurrence relation. However, since we are interested in asymptotic running times, we do not need the precise solution of a "normal" recurrence relation as described above. A normal recurrence relation defines a sequence of numbers. If the time complexity of an algorithm as a worst-case analysis was given by a normal recurrence relation then the number sequence a_1, a_2, a_3, ..., defined by that relation, would describe the running time of algorithm precisely, *i.e.* for input of size n, the maximum number of steps the algorithm makes over all inputs of size n is precisely a_n. We do not need such a precise analysis and often it is impossible to derive one. So, the recurrence relations we use when analysing an algorithm typically have bases $\Theta(1)$, for example:

$$T(n) = T(n-1)+1,\ n \geq 2 \tag{3.2}$$
$$T(1) = \Theta(1)$$

Infinitely many number sequences are solutions to (3.2). To solve such a recurrence relation means to find the asymptotic growth of any of those sequences. The best solution we can hope for, asymptotically, is the one given by the Θ notation. If we are unable to pin down the asymptotic growth in that sense, our second best option is to find functions $f(n)$ and $g(n)$, such that $f(n) = o(g(n))$ and $T(n) = \Omega(f(n))$ and $T(n) = O(g(n))$. The best solution for the recurrence relation (3.2), in the asymptotic sense, is $T(n) = \Theta(n)$. Another solution, not as good as this one, is, for example, $T(n) = \Omega(\sqrt{n})$ and $T(n) = O(n^2)$.

In the problems that follow, we distinguish the two types of recurrence relation by the initial conditions. If the initial condition is given by a precise expression as in (3.1) we have to give a precise answer such as $T(n) = n$, and if the initial condition is $\Theta(1)$ as in (3.2) we want only the growth rate.

It is possible to omit the initial condition altogether in the description of the recurrence. If we do so we assume tacitly the initial condition is $T(c) = \Theta(1)$ for some positive constant c. The reason to do that may be that it is pointless to specify the usual $T(1)$; however, it may be the case that the variable never reaches value one. For instance, consider the recurrence relation

$$T(n) = T\left(\left\lfloor \frac{n}{2} \right\rfloor + 17\right) + n$$

which we solve below (Problem 41 on page 42). To specify "$T(1) = \Theta(1)$" for it is *wrong*.

3.1.1 Iterators

The recurrence relations can be partitioned into the following two classes, assuming T is the function of the recurrence relations as above.

1. The ones in which T appears only once on the right-hand side as in (3.1).

2. The ones in which T appears mutiple times on the right-hand side, for instance:

$$T(n) = T(n-1) + T(n-2) + T(n-3) + n \tag{3.3}$$

We will call them relations with *single occurrence* and with *multiple occurrences*, respectively. We find it helpful to make that distinction because in general only the relations with single occurrence are ameaneable to the method of unfolding (see below). If the relation is with single occurrence we define *the iterator* of the relation as the iterative expression that shows how the variable decreases. For example, the iterator of (3.1) is:

$$n \to n-1 \tag{3.4}$$

It is not practical to define iterators for relations with multiple occurrences. If we wanted to define iterators for them as well, they would have a set of functions on the right-hand side, for instance the iterator of (3.3) would be

$$n \to \{n-1, n-2, n-3\}$$

and that does help the analysis of the relation. So, we define iterators only for relations with single occurrence. The iterators that are easiest to deal with (and, fortunately, occur often in practice) are the ones in which the function on the right-hand side is subtraction or division (by constant > 1):

$$n \to n - c,\ c > 0 \tag{3.5}$$

$$n \to \frac{n}{b},\ b > 1 \tag{3.6}$$

Another possibility is that function to be some root of n:

$$n \to \sqrt[d]{n},\ d > 1 \tag{3.7}$$

Note that the direction of the assignment in the iterator is the opposite to the one in the recurrence relation (compare (3.1) with (3.4)). The reason is that a recurrence has to phases: descending and ascending. In the descending phase we start with some value n for the variable and decrease it in successive steps till we reach the initial condition; in the ascending phase we go back from the initial condition "upwards". The left-to-right direction of the iterator refers to the descending phase, while the right-to-left direction of the assignment in the recurrence refers to the ascending phase.

It is important to be able to estimate the number of times an iterator will be executed before its variable becomes 1 (or whatever value the initial conditions specify). If the variable n is integer, the iterator $n \to n-1$ is the most basic one we can possibly have. The number of times it is executed before n becomes any *a priori* fixed constant is $\Theta(n)$. That has to be obvious. Now consider (3.5). We ask the same question: how many times it is executed before n becomes a constant. Substitute n by cm and (3.5) becomes:

$$cm \to c(m-1) \tag{3.8}$$

The number of times (3.8) is executed (before m becomes a constant) is $\Theta(m)$. Since $m = \Theta(n)$, we conclude that (3.5) is executed $\Theta(n)$ times.

Consider the iterator (3.6). To see how many times it is executed before n becomes a constant (fixed *a priori*)) can be estimated as follows. Substitute n by b^m and (3.6) becomes

$$b^m \to b^{m-1} \tag{3.9}$$

(3.9) is executed $\Theta(m)$ times because $m \to m - 1$ is executed $\Theta(m)$ times. Since $m = \log_b n$, we conclude that (3.6) is executed $\Theta(\log_b n)$ times, *i.e.* $\Theta(\lg n)$ times. We see that the concrete value of b is immaterial with respect to the asymptotics of the number of executions, provided $b > 1$.

Now consider (3.7). To see how many times it is executed before n becomes a constant, substitute n by d^{d^m}. (3.7) becomes

$$d^{d^m} \to d^{\frac{d^m}{d}} = d^{d^{m-1}} \tag{3.10}$$

(3.10) is executed $\Theta(m)$ times. As $m = \log_d \log_d n$, we conclude that (3.7) is executed $\Theta(\log_d \log_d n)$ times, *i.e.* $\Theta(\lg \lg n)$ times. Again we see that the value of the constant in the iterator, namely d, is immaterial as long as $d > 1$.

Let us consider an iterator that decreases even faster than (3.7):

$$n \to \lg n \tag{3.11}$$

The number of times it is executed before n becomes a constant is $\lg^* n$, which follows right from Definion 1 on page 11.

Let us summarise the rates of decrease of the iterators we just considered assuming the mentioned "constants of decrease" b and d are 2.

iterator	*asymptotics of the number executions*	*alternative form (see Definion 1)*
$n \to n - 1$	n	$\lg^{(0)} n$
$n \to n/2$	$\lg n$	$\lg^{(1)} n$
$n \to \sqrt{n}$	$\lg \lg n$	$\lg^{(2)} n$
$n \to \lg n$	$\lg^* n$	$\lg^* n$

There is a gap in the table. One would ask, what is the function $f(n)$, such that the iterator $n \to f(n)$ is executed, asymptotically, $\lg \lg \lg n$ times, *i.e.* $\lg^{(3)} n$ times. To answer that question, consider that $f(n)$ has to be such that if we substitute n by 2^m, the number of executions is the same as in the iterator $m \to \sqrt{m}$. But $m \to \sqrt{m}$ is the same as $\lg n \to \sqrt{\lg n}$, *i.e.* $n \to 2^{\sqrt{\lg n}}$. We conclude that $f(n) = 2^{\sqrt{\lg n}}$. To check this, consider the iterator

$$n \to 2^{\sqrt{\lg n}} \tag{3.12}$$

Substitute n by $2^{2^{2^m}}$ in (3.12) to obtain:

$$2^{2^{2^m}} \to 2^{\sqrt{\lg 2^{2^{2^m}}}} = 2^{\sqrt{2^{2^m}}} = 2^{2^{\frac{2^m}{2}}} = 2^{2^{2^{m-1}}} \tag{3.13}$$

Clearly, (3.13) is executed $m = \lg \lg \lg n = \lg^{(3)} n$ times.

A further natural question is, what the function $\phi(n)$ is, such that the iterator $n \to \phi(n)$ is executed $\lg^{(4)} n$ times. Applying the reasoning we used to derive $f(n)$, $\phi(n)$ has to be such that if we substitute n by 2^m, the number of executions is the same as in $m \to 2^{\sqrt{\lg m}}$. As $m = \lg n$, the latter becomes $\lg n \to 2^{\sqrt{\lg \lg n}}$, *i.e.* $n \to 2^{2^{\sqrt{\lg \lg n}}}$. So, $\phi(n) = 2^{2^{\sqrt{\lg \lg n}}}$. We can fill in two more rows in the table:

iterator	*asymptotics of the number executions*	*alternative form (see Definion 1)*
$n \to n - 1$	n	$\lg^{(0)} n$
$n \to n/2$	$\lg n$	$\lg^{(1)} n$
$n \to \sqrt{n}$	$\lg \lg n$	$\lg^{(2)} n$
$n \to 2^{\sqrt{\lg n}}$	$\lg \lg \lg n$	$\lg^{(3)} n$
$n \to 2^{2^{\sqrt{\lg \lg n}}}$	$\lg \lg \lg \lg n$	$\lg^{(4)} n$
$n \to \lg n$	$\lg^* n$	$\lg^* n$

Let us define, analogously to Definion 1, the function base-two iterated exponent.

Definition 3 (iterated exponent)**.** *Let* i *be a nonnegative integer.*

$$\text{itexp}^{(i)}(n) = \begin{cases} n, & \text{if } i = 0 \\ 2^{\text{itexp}^{(i-1)}(n)}, & \text{if } i > 0 \end{cases}$$ □

Having in mind the results in the table, we conjecture, and it should not be too difficult to prove by induction, that the iterator:

$$n \to \text{itexp}^{(k)}\left(\sqrt{\lg^{(k)} n}\right) \tag{3.14}$$

is executed $\lg^{(k+2)} n$ times for $k \in \mathbb{N}$.

3.1.2 Recursion trees

Assume we are given a recurrence relation of the form:

$$T(n) = k_1 T(f_1(n)) + k_2 T(f_2(n)) + \ldots + k_p T(f_p(n)) + \phi(n) \tag{3.15}$$

where k_i, $1 \le i \le p$ are positive integer constants, $f_i(n)$ for $1 \le i \le p$ are integer-valued functions such that $n > f(n)$ for all $n \ge n_0$ where n_0 is the largest (constant) value of the argument in any initial condition, and $\phi(n)$ is some positive function. It is not necessary $\phi(n)$ to be positive as the reader will see below; however, if $T(n)$ describes the running time of a recursive algorithm then $\phi(n)$ has to be positive. We build a special kind of rooted tree that corresponds to our recurrence relation. Each node of the tree corresponds to one particular value of the variable that appears in the process of unfolding the relation, the value that corresponds to the root being n. That value we call *the level* of the node. Further, with each node we associate $\phi(m)$ where m is the level of that node. We call that, *the cost* of the node. Further, each node—as long as no initial condition has been reached yet—has $k_1 + k_2 + \ldots + k_p$ children, k_i of them being at level defined by f_i for $1 \le i \le p$. For example, if our recurrence is

$$T(n) = 2T(n-1) + n^2$$

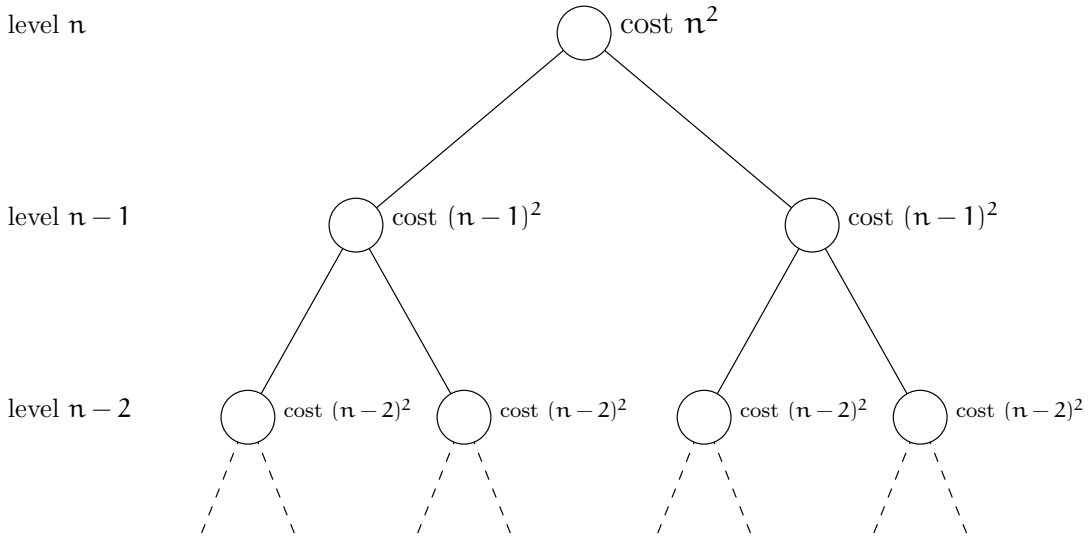

Figure 3.1: The recursion tree of $T(n) = 2T(n-1) + n^2$.

the recursion tree is as shown on Figure 3.1. It is a complete binary tree. It is binary because there are two invocations on the right side, *i.e.* $k_1 + k_2 + \ldots + k_p = 2$ in the above terminology. And it is complete because it is a recurrence with a single occurrence. Note that if $k_1 + k_2 + \ldots + k_p$ equals 1 then the recursion tree degenerates into a path.

The size of the tree depends on n so we can not draw the whole tree. The figure is rather a suggestion about it. The bottom part of the tree is missing because we have not mentioned the initial conditions. The solution of the recursion—and that is the goal of the tree, to help us solve the recursion—is the total sum of all the costs. Typically we sum by levels, so in the current example the sum will be

$$n^2 + 2(n-1)^2 + 4(n-2)^2 + \ldots$$

The general term of this sum is $2^k(n-k)^2$. The "..." notation hides what happens at the right end, however, we agreed the initial condition is for some, it does not matter, what constant value of the variable. Therefore, the sum

$$\sum_{k=0}^{n} 2^k(n-k)^2$$

has the same growth rate as our desired solution. Let us find a closed form for that sum.

$$\sum_{k=0}^{n} 2^k(n-k)^2 = n^2 \sum_{k=0}^{n} 2^k - 2n \sum_{k=0}^{n} 2^k k + \sum_{k=0}^{n} 2^k k^2$$

Having in mind Problem 73 on page 80 and Problem 74 on page 80, that expression becomes

$$n^2(2^{n+1} - 1) - 2n((n-1)2^{n+1} + 2) + n^2 2^{n+1} - 2n2^{n+1} + 4.2^{n+1} - 6 =$$
$$n^2.2^{n+1} - n^2 - 2n(n.2^{n+1} - 2^{n+1} + 2) + n^2 2^{n+1} - 2n2^{n+1} + 4.2^{n+1} - 6 =$$
$$2.n^2.2^{n+1} - n^2 - 2.n^2.2^{n+1} + 2n.2^{n+1} - 4n - 2n2^{n+1} + 4.2^{n+1} - 6 =$$
$$4.2^{n+1} - n^2 - 4n - 6$$

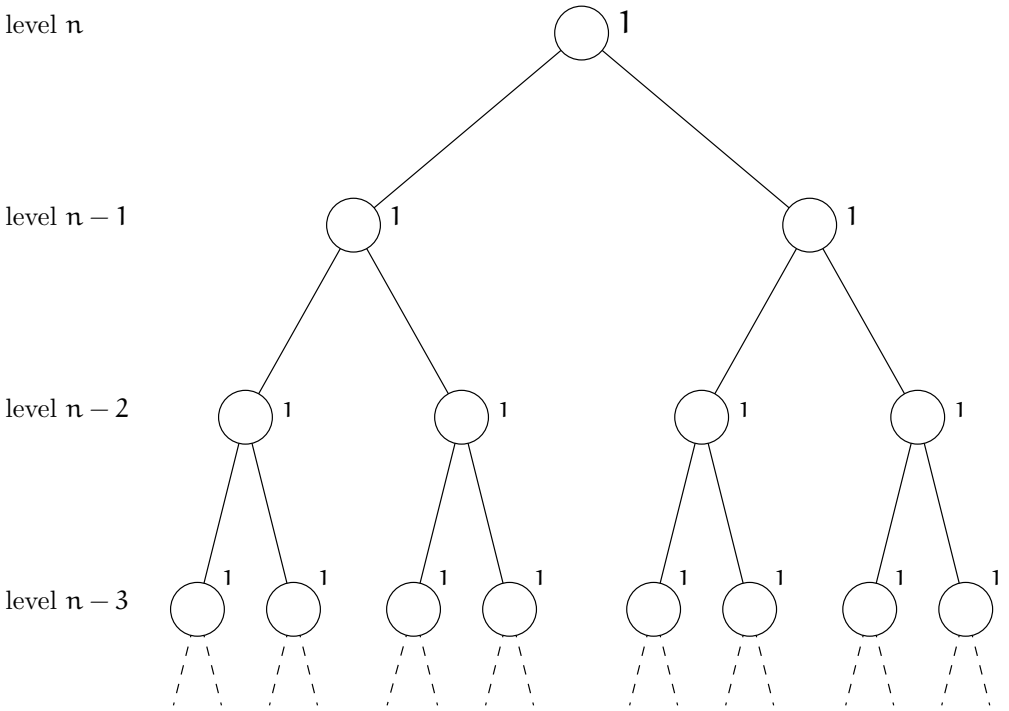

Figure 3.2: The recursion tree of $T(n) = 2T(n-1) + 1$.

It follows that $T(n) = \Theta(2^n)$.

The correspondence between a recurrence relation and its recursion tree is not necessarily one-to-one. Consider the recurrence relation (3.2) on page 34 and its recursion tree (Figure 3.2). The cost at level n is 1, at level $n-1$ is 2, at level $n-2$ is 4, at level $n-3$ is 8, *etc.* The tree is obviously complete. Let us now rewrite (3.2) as follows.

$$\begin{aligned} T(n) &= 2T(n-1) + 1 \;\Leftrightarrow\; T(n) = T(n-1) + T(n-1) + 1 \\ T(n-1) &= 2T(n-2) + 1 \\ T(n) &= T(n-1) + 2T(n-2) + 2 \end{aligned}$$

We have to alter the initial conditions for this rewrite, adding $T(2) = 3$. Overall the recurrence becomes

$$\begin{aligned} T(n) &= T(n-1) + 2T(n-2) + 2 \\ T(2) &= 3 \\ T(1) &= 1 \end{aligned} \tag{3.16}$$

Recurrences (3.2) and (3.16) are equivalent. One can say these are different ways of writing down the same recurrence because both of them define one and the same sequence, namely $1, 3, 7, 15, \ldots$ However, their recursion trees are neither the same nor isomorphic. Figure 3.3 shows the tree of (3.16). To give a more specific example, Figure 3.4 shows the recursion tree of (3.16) for $n = 5$. It shows the whole tree, not just the top, because the variable has a concrete value. Therefore the initial conditions are taken into account. The reader can

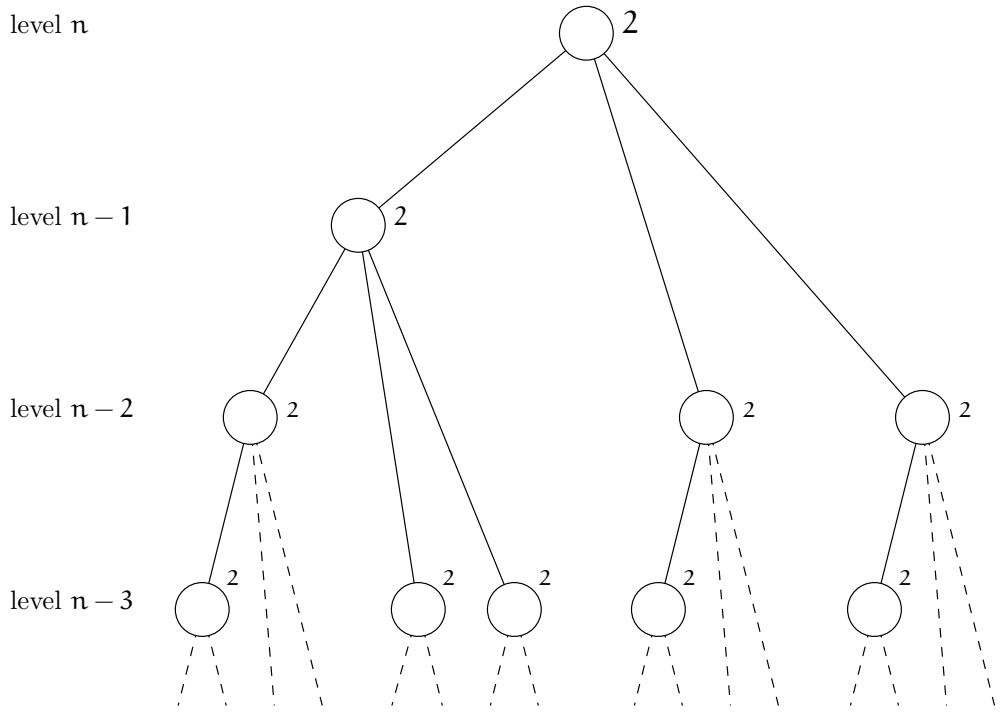

Figure 3.3: The recursion tree of $T(n) = T(n-1) + 2T(n-2) + 2$.

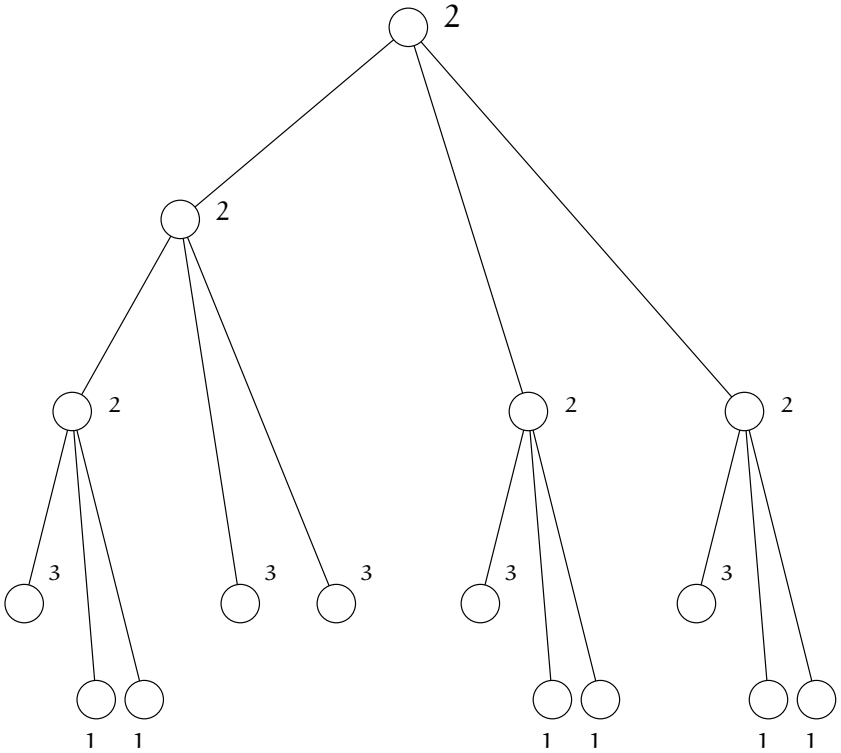

Figure 3.4: The recursion tree of $T(n) = T(n-1) + 2T(n-2) + 2$, $T(2) = 3$, $T(1) = 1$, for $n = 5$.

easily see the total sum of the costs over the tree from Figure 3.4 is 31, the same as the tree from Figure 3.2 for $n = 5$. However, the sum 31 on Figure 3.2 is obtained as $1+2+4+8+16$, if we sum by levels. In the case with Figure 3.4 we do not have obvious definition of levels.

- If we define the levels as the vertices that have the same value of the variable, we have 5 levels and the sum is derived, level-wise, as $2 + 2 + 6 + 15 + 6 = 31$.
- If we define the levels as the vertices that are at the same distance to the root, we have only 4 levels and the sum is derived, level-wise, as $2 + 6 + 18 + 5 = 31$.

Regardless of how we define the levels, the derivation is not $1 + 2 + 4 + 8 + 16$.

3.2 Problems

Our repertoire of methods for solving recurrences is:

- by induction,
- by unfolding,
- by considering the recursion tree,
- by the Master Theorem, and
- by the method of the characteristic equation.

3.2.1 Induction, unfolding, recursion trees

Problem 39. *Solve*

$$\begin{aligned} T(n) &= 2T(n-1) + 1 \\ T(0) &= 0 \end{aligned} \tag{3.17}$$

Solution:
We guess that $T(n) = 2^n - 1$ for all $n \geq 1$ and prove it by induction on n.

Basis: $n = 1$. We have $T(1) = 2T(0) + 1$ by substituting n with 1. But $T(0) = 0$, thus $T(1) = 2 \times 0 + 1 = 1$. On the other hand, substituting n with 1 in our guessed solution, we have $2^1 - 1 = 1$.

Inductive hypothesis: assume $T(n) = 2^n - 1$ for some $n > 1$.

Inductive step: $T(n+1) = 2T(n)+1$ by definition. Apply the inductive hypothesis to obtain $T(n + 1) = 2(2^n - 1) + 1 = 2^{n+1} - 1$. □

The proofs by induction have one major drawback – making a good guess can be a form of art. There is no recipe, no algorithm for making a good guess in general. It makes sense to compute several initial values of the sequence defined by the recurrence and try to see a pattern in them. In the last problem, $T(1) = 1$, $T(2) = 3$, $T(3) = 7$ and it is reasonable to assume that $T(n)$ is $2^n - 1$. Actually, if we think about (3.17) in terms of the binary representation of $T(n)$, it is pretty easy to spot that (3.17) performs a shift-left by one

position and then turns the least significant bit from 0 into 1. As we start with $T(1) = 1$, clearly

$$T(n) = \underbrace{1\,1\,1\,\ldots\,1}_{n \text{ times}}{}_b$$

For more complicated recurrence relations, however, seeing a pattern in the initial values of the sequence, and thus making a good guess, can be quite challenging. If one fails to see such a pattern it is a good idea to check if these numbers are found in *The On-Line Encyclopedia of Integer Sequences* [Slo]. Of course, this advice is applicable when we solve precise recurrence relations, not asymptotic ones.

Problem 40. *Solve*

$$T(n) = T(n-1) + n \tag{3.18}$$
$$T(0) = 1$$

Solution:
By unfolding (also called unwinding) of the recurrence down to the initial condition.

$$\begin{aligned}
T(n) &= T(n-1) + n \quad \text{directly from (3.18)} \\
&= T(n-2) + n - 1 + n \quad \text{substitute } n \text{ with } n-1 \text{ in (3.18)} \\
&= T(n-3) + n - 2 + n - 1 + n \quad \text{substitute } n-1 \text{ with } n-2 \text{ in (3.18)} \\
&\ldots \\
&= T(0) + 1 + 2 + 3 + \ldots + n - 2 + n - 1 + n = \\
&= 1 + 1 + 2 + 3 + \ldots + n - 2 + n - 1 + n = \\
&= 1 + \frac{n(n+1)}{2}
\end{aligned}$$

This method is considered to be not as formally precise as the induction. The reason is that we inevitably skip part of the derivation—the dot-dot-dot "..." part—leaving it to the imagination of the reader to verify the derived closed formula. Problem 40 is trivially simple and it is certain beyond any doubt that if we start with $T(n-3) + n - 2 + n - 1 + n$ and systematically unfold $T(i)$, decrementing by one values of i, eventually we will "hit" the initial condition $T(0)$ and the "tail" will be $1 + 2 + 3 + \ldots + n - 2 + n - 1 + n$. The more complicated the expression is, however, the more we leave to the imagination of the reader when unfolding.

One way out of that is to use the unfolding to derive a closed formula and then prove it by induction. □

Problem 41. *Solve*

$$T(n) = 2T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n \tag{3.19}$$
$$T(1) = \Theta(1) \tag{3.20}$$

Solution:
We prove that $T(n) = \Theta(n \lg n)$ by induction on n. To accomplish that we prove separately that $T(n) = O(n \lg n)$ and $T(n) = \Omega(n \lg n)$.

Part I: Proof that $T(n) = O(n \lg n)$, that is, there exists a positive constant c and some n_0, such that for all $n \geq n_0$,

$$T(n) \leq cn \lg n \tag{3.21}$$

There is a potential problem with the initial condition because for $n = 1$ the right-hand side of (3.21) becomes $c.1. \lg 1 = 0$, and $0 \neq \Theta(1)$. However, it is easy to deal with that issue, just do not take $n = 1$ as basis. Taking $n = 2$ as basis works as $c.2. \lg 2$ is not zero. However, note that $n = 2$ is not sufficient basis! There are certain values for n, for example 3, such that the iterator of this recurrence, namely

$$n \to \left\lfloor \frac{n}{2} \right\rfloor$$

"jumps over" 2, having started from one of them. Indeed, $\left\lfloor \frac{3}{2} \right\rfloor = 1$, therefore the iterator, starting from 3, does

$$3 \to 1$$

and then goes infinite descent. The solution is to take two bases, for both $n = 2$ and $n = 3$. It is certain that no matter what n is the starting one, the iterator will at one moment "hit" either 2 or 3. So, the bases of our proof are:

$$T(2) = \Theta(1) \tag{3.22}$$
$$T(3) = \Theta(1) \tag{3.23}$$

Of course, that does not say that $T(2) = T(3)$, it says there exist constants c_2 and c_3, such that:

$$c_2 \leq c.2 \lg 2$$
$$c_3 \leq c.3 \lg 3$$

Our induction hypothesis is that relative to some sufficiently large n, (3.21) holds for some positive constant c all values of the variable between 3 and n, excluding n. The induction step is to prove (3.21), using the hypothesis. So,

$$\begin{aligned}
T(n) &= 2T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n \qquad \text{this is the defintion of } T(n) \\
&\leq 2.c. \left\lfloor \frac{n}{2} \right\rfloor \lg \left\lfloor \frac{n}{2} \right\rfloor + n \qquad \text{from the inductive hypothesis} \\
&\leq 2.c. \frac{n}{2} \lg \frac{n}{2} + n \\
&= cn(\lg n - 1) + n \\
&= cn \lg n + (1 - c)n \qquad (3.24) \\
&\leq cn \lg n \qquad \text{provided that } (1 - c) \leq 0 \Leftrightarrow c \geq 1 \qquad (3.25)
\end{aligned}$$

If $c \geq 1$, the proof is valid. If we want to be perfectly precise we have to consider the two bases as well to find a value for c that works. Namely,

$$c = \max\left\{1, \frac{c_2}{2\lg 2}, \frac{c_3}{3\lg 3}\right\}$$

In our proofs from now on we will not consider the initial conditions when choosing an appropriate constant.

Part II: Proof that $T(n) = \Omega(n \lg n)$, that is, there exists a positive constant d and some n_1, such that for all $n \geq n_1$,

$$T(n) \geq dn \lg n \tag{3.26}$$

We will ignore the basis of the induction and focus on the hypothesis and the inductive step only. Applying the inductive hypothesis to (3.26), we get:

$$\begin{aligned}
T(n) &\geq 2d \left\lfloor \frac{n}{2} \right\rfloor \lg \left\lfloor \frac{n}{2} \right\rfloor + n \qquad \text{from the inductive hypothesis} \\
&\geq 2d \left(\frac{n}{2} - 1\right) \lg \left\lfloor \frac{n}{2} \right\rfloor + n \\
&= d(n-2) \lg \left\lfloor \frac{n}{2} \right\rfloor + n \\
&\geq d(n-2) \lg \left(\frac{n}{4}\right) + n \\
&= d(n-2)(\lg n - 2) + n \\
&= dn \lg n + n(1-2d) - 2d \lg n + 4d \\
&\geq dn \lg n \qquad \text{provided that } n(1-2d) - 2d \lg n + 4d \geq 0
\end{aligned}$$

So (3.26) holds when $n(1-2d) - 2d \lg n + 4d \geq 0$ is nonnegative. Observe that for $d = \frac{1}{4}$ the inequality becomes

$$\frac{n}{2} + 16 \geq \frac{1}{2} \lg n$$

It certainly holds $\forall n \geq 2$, therefore the choice $d = \frac{1}{4}$ and $n_1 = 2$ suffices for our proof. □

Problem 42. *Solve*

$$T(n) = 2T\left(\left\lceil \frac{n}{2} \right\rceil\right) + n \tag{3.27}$$

$$T(1) = \Theta(1) \tag{3.28}$$

Solution:
We prove that $T(n) = \Theta(n \lg n)$ by induction on n. To accomplish that we prove separately that $T(n) = O(n \lg n)$ and $T(n) = \Omega(n \lg n)$. We ignore the basis of the induction – the solution of Problem 41 gives us enough confidence that we can handle the basis if we wanted to.

Part I: Proof that $T(n) = O(n \lg n)$, that is, there exists a positive constant c and some n_0, such that for all $n \geq n_0$,

$$T(n) \leq cn \lg n \tag{3.29}$$

From the inductive hypothesis

$$\begin{aligned}
T(n) &\leq 2.c.\left\lceil \frac{n}{2} \right\rceil \lg \left\lceil \frac{n}{2} \right\rceil + n \\
&\leq 2.c.\left(\frac{n}{2}+1\right) \lg \left\lceil \frac{n}{2} \right\rceil + n \\
&\leq 2.c.\left(\frac{n}{2}+1\right) \lg \left(\frac{3n}{4}\right) + n \qquad \text{because } \frac{3n}{4} \geq \left\lceil \frac{n}{2} \right\rceil \ \forall n \geq 2 \\
&= c(n+2)(\lg n + \lg 3 - 2) + n \\
&= cn \lg n + cn(\lg 3 - 2) + 2c \lg n + 2c(\lg 3 - 2) + n \\
&\leq cn \lg n \qquad \text{if } \ cn(\lg 3 - 2) + 2c \lg n + 2c(\lg 3 - 2) + n \leq 0
\end{aligned}$$

Consider

$$cn(\lg 3 - 2) + 2c \lg n + 2c(\lg 3 - 2) + n = (c(\lg 3 - 2) + 1)n + 2c \lg n + 2c(\lg 3 - 2)$$

Its asymptotic growth rate is determined by the linear term. If the constant $c(\lg 3 - 2) + 1$ is negative then the whole expression is certainly negative for all sufficiently large values of n. In other words, for the sake of brevity we do not specify precisely what n_0 is. In order to have $c(\lg 3 - 2) + 1 < 0$ it must be the case that $c > \frac{1}{2-\lg 3}$. So, any $c > \frac{1}{2-\lg 3}$ works for our proof.

Part II: Proof that $T(n) = \Omega(n \lg n)$, that is, there exists a positive constant d and some n_1, such that for all $n \geq n_1$,

$$T(n) \geq dn \lg n \tag{3.30}$$

From the inductive hypothesis

$$\begin{aligned}
T(n) &\geq 2.d.\left\lceil \frac{n}{2} \right\rceil \lg \left\lceil \frac{n}{2} \right\rceil + n \\
&\geq 2.d.\left(\frac{n}{2}\right) \lg \left(\frac{n}{2}\right) + n \\
&= dn(\lg n - 1) + n \\
&= dn \lg n + (1 - d)n \\
&\geq dn \lg n \qquad \text{provided that } (1 - d)n \geq 0
\end{aligned} \tag{3.31}$$

It follows that any d such that $0 < d \leq 1$ workds for our proof. □

❣❣ **NB** ❣❣ As explained in [CLR00, pp. 56–57], it is easy to make a wrong "proof" of the growth rate by induction if one is not careful. Suppose one "proves" the solution of (3.19) is $T(n) = O(n)$ by first guessing (incorrectly) that $T(n) \leq cn$ for some positive constant c and then arguing

$$\begin{aligned}
T(n) &\leq 2c\left\lfloor \frac{n}{2} \right\rfloor + n \\
&\leq cn + n \\
&= (c + 1)n \\
&= O(n)
\end{aligned}$$

While it is certainly true that $cn + n = O(n)$, that is irrelevant to the proof. The proof started relative to the constant c and has to finish relative to it. In other words, the proof has to show that $T(n) \leq cn$ for the choice of c in the inductive hypothesis, not that $T(n) \leq dn$ for some positive constant d which is not c. Proving that $T(n) \leq (c+1)n$ does *not* constitute a proof of the statement we are after.

Problem 43. *Solve*

$$T(n) = T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + 1 \tag{3.32}$$

$$T(1) = \Theta(1) \tag{3.33}$$

Solution:
We prove that $T(n) = \Theta(\lg n)$ by induction on n. To accomplish that we prove separately that $T(n) = O(\lg n)$ and $T(n) = \Omega(\lg n)$.

Part I: Proof that $T(n) = O(\lg n)$, that is, there exists a positive constant c and some n_0, such that for all $n \geq n_0$,

$$T(n) \leq c \lg n \tag{3.34}$$

By the inductive hypothesis,

$$\begin{aligned} T(n) &\leq c \lg\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + 1 \\ &\leq c \lg\left(\frac{3n}{4}\right) + 1 \qquad \text{since } \frac{3n}{4} \geq \left\lfloor \frac{n}{2} \right\rfloor \text{ for all sufficiently large } n \\ &= c(\lg n + \lg 3 - 2) + 1 \\ &= c \lg n + c(\lg 3 - 2) + 1 \\ &\leq c \lg n \qquad \text{provided that } c(\lg 3 - 2) + 1 \leq 0 \Leftrightarrow c \geq \frac{1}{2 - \lg 3} \end{aligned}$$

Part II: Proof that $T(n) = \Omega(\lg n)$, that is, there exists a positive constant d and some n_1, such that for all $n \geq n_1$,

$$T(n) \geq d \lg n \tag{3.35}$$

By the inductive hypothesis,

$$\begin{aligned} T(n) &\geq d \lg\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + 1 \\ &\geq d \lg\left(\frac{n}{4}\right) + 1 \qquad \text{since } \frac{n}{4} \leq \left\lfloor \frac{n}{2} \right\rfloor \text{ for all sufficiently large } n \\ &= d \lg n - 2d + 1 \qquad \text{provided that } -2d + 1 \geq 0 \Leftrightarrow d \leq \frac{1}{2} \end{aligned}$$

□

Problem 44. *Solve*

$$T(n) = 2T\left(\left\lfloor \frac{n}{2} + 17 \right\rfloor\right) + n \tag{3.36}$$

$$\tag{3.37}$$

Solution:
We prove that $T(n) = \Theta(n \lg n)$ by induction on n. To accomplish that we prove separately that $T(n) = O(\lg n)$ and $T(n) = \Omega(\lg n)$. Note that the initial condition is *not* $T(1) = \Theta(1)$ in this problem because the itetor

$$n \to \left\lfloor \frac{n}{2} \right\rfloor + 17$$

never reaches 1 when starting from any sufficiently large n. Its fixed point is 34 but we avoid mentioning the awkward initial condition $T(34) = \Theta(1)$.

Part I: Proof that $T(n) = O(n \lg n)$, that is, there exists a positive constant c and some n_0, such that for all $n \geq n_0$,

$$T(n) \leq cn \lg n \tag{3.38}$$

By the inductive hypothesis,

$$\begin{aligned} T(n) &\leq 2c \left(\left\lfloor \frac{n}{2} \right\rfloor + 17 \right) \lg \left(\left\lfloor \frac{n}{2} \right\rfloor + 17 \right) + n \\ &= 2c \left(\frac{n}{2} + 17 \right) \lg \left(\frac{n}{2} + 17 \right) + n \\ &= c(n+34) \lg \left(\frac{n+34}{2} \right) + n \\ &= c(n+34) \big(\lg (n+34) - 1 \big) + n \\ &\leq c(n+34) \big(\lg (\sqrt{2}n) - 1 \big) + n \end{aligned} \tag{3.39}$$

because for all sufficiently large values of n, say $n \geq 100$, it is the case that $\sqrt{2}n \geq n + 34$.

$$\begin{aligned} T(n) &\leq c(n+34) \big(\lg (\sqrt{2}n) - 1 \big) + n \qquad \text{from (3.39)} \\ &= c(n+34) \left(\lg n + \frac{1}{2} \lg 2 - 1 \right) + n \\ &= c(n+34) \left(\lg n - \frac{1}{2} \right) + n \\ &= cn \lg n + 34c \lg n - \frac{cn}{2} - 17c + n \\ &\leq cn \lg n \qquad \text{provided that } 34c \lg n - \frac{cn}{2} - 17c + n \leq 0 \end{aligned}$$

In order $34c \lg n - \frac{cn}{2} - 17c + n = n\left(1 - \frac{c}{2}\right) + 34c \lg n - 17c$ to be non-positive for all sufficiently large n it suffices $\left(1 - \frac{c}{2}\right)$ to be negative because the linear function dominated the logarithmic function. A more detailed analysis is the following. Fix $c = 4$. The expression becomes $(-1)n + 136 \lg n - 136$.

$$(-1)n + 136 \lg n - 136 \leq 0 \;\Leftrightarrow\; n \geq 136(\lg n - 1) \;\Leftrightarrow\; \frac{n}{\lg n - 1} \geq 136$$

For $n = 65536 = 2^{16}$ the inequality holds:

$$\frac{2^{16}}{15} \geq 136$$

so we can finish the proof with choosing $n_0 = 65536$ and $c = 4$.

Part II: Proof that $T(n) = \Omega(n \lg n)$, that is, there exists a positive constant d and some n_1, such that for all $n \geq n_1$,

$$T(n) \geq dn \lg n$$

By the inductive hypothesis,

$$\begin{aligned} T(n) &\geq 2d\left(\left\lfloor \frac{n}{2} \right\rfloor + 17\right) \lg\left(\left\lfloor \frac{n}{2} \right\rfloor + 17\right) + n \\ &\geq 2d\left(\left\lfloor \frac{n}{2} \right\rfloor\right) \lg\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n \\ &\geq 2d\left(\frac{n}{2}\right) \lg\left(\frac{n}{2}\right) + n \\ &= dn(\lg n - 1) + n \\ &= dn \lg n + (1-d)n \\ &\geq dn \lg n \qquad \text{provided that } 1 - d \geq 0 \Leftrightarrow d \leq 1 \end{aligned}$$ □

Problem 45. *Solve*

$$\begin{aligned} T(n) &= 2T\left(\frac{n}{2}\right) + 1 \\ T(1) &= \Theta(1) \end{aligned} \tag{3.40}$$

by the method of the recursion tree.

Solution:
The recursion tree is shown on Figure 3.5. The solution is the sum over all levels:

$$T(n) = \underbrace{1 + 2 + 4 + 8 + \ldots}_{\text{the number of terms is the height of the tree}} \tag{3.41}$$

The height of the tree is the number of times the iterator

$$n \to \frac{n}{2}$$

is executed before the variable becomes 1. As we already saw, that number is $\lg n$[†]. So, (3.41) in fact is

$$\begin{aligned} T(n) &= \underbrace{1 + 2 + 4 + 8 + \ldots}_{\lg n \text{ terms}} = 1 + 2 + 4 + 8 + \ldots + \frac{n}{2} + n \\ &= \sum_{i=1}^{\lg n} \frac{n}{2^i} = n\left(\sum_{i=1}^{\lg n} \frac{1}{2^i}\right) \leq n \underbrace{\left(\sum_{i=1}^{\infty} \frac{1}{2^i}\right)}_{2} = 2n \end{aligned}$$

We conclude that $T(n) = \Theta(n)$. □

[†]Actually it is $\lfloor \lg n \rfloor$ but that is immaterial with respect to the asymptotic growth of $T(n)$.

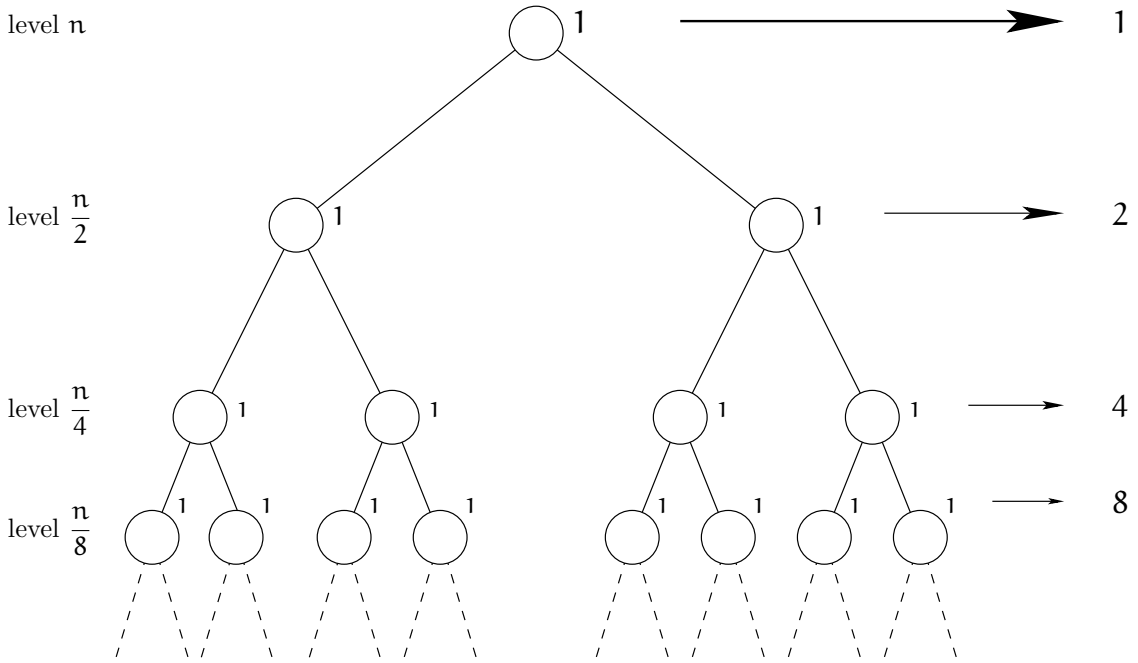

Figure 3.5: The recursion tree of $T(n) = 2T\left(\frac{n}{2}\right) + 1$.

However, that proof by the method of the recursion tree can be considered insufficiently precise because it involves several approximations and the use of imagination—the dot-dot-dot notations. Next we demonstrate a proof by induction of the same result. We may think of the proof with recursion tree as a mere way to derive a good guess to be verified formally by induction.

Problem 46. *Prove by induction on* n *that the solution to*

$$T(n) = 2T\left(\frac{n}{2}\right) + 1 \tag{3.42}$$
$$T(1) = \Theta(1)$$

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

Solution:
We prove separately that $T(n) = O(n)$ and $T(n) = \Omega(n)$.

Part I: Proof that $T(n) = O(n)$. For didactic purposes we will first make an unsuccessful attempt.

Part I, try 1: Assume there exists a positive constant c and some n_0, such that for all $n \geq n_0$,

$$T(n) \leq cn \tag{3.43}$$

By the inductive hypothesis,

$$\begin{aligned} T(n) &\leq 2c\frac{n}{2} + 1 \\ &= cn + 1 \end{aligned}$$

Our proof ran into a problem: no matter what positive c we choose, it is not true that $cn + 1 \leq cn$, and thus (3.43) cannot be shown to hold. Of course, that failure does *not* mean our claim $T(n) = \Theta(n)$ is false. It simply means that (3.43) is inappropriate. We amend the situation by a technique known as *strenthening the claim*. It consists of stating an appropriate claim that is stronger than (3.43) and then proving it by induction. Intuitively, that stronger claim has to contain some minus sign in such a way that after applying the inductive hypothesis, there is a term like $-c$ that can "cope with" the $+1$.

Part I, try 2: Assume there exists positive constants b and c and some n_0, such that for all $n \geq n_0$,

$$T(n) \leq cn - b \tag{3.44}$$

By the inductive hypothesis,

$$\begin{aligned} T(n) &\leq 2\left(c\frac{n}{2} - b\right) + 1 \\ &= cn - 2b + 1 \\ &\leq cn \qquad \text{for any } b \text{ such that } -2b + 1 \leq 0 \Leftrightarrow b \geq \frac{1}{2} \end{aligned}$$

Part II: Proof that $T(n) = \Omega(n)$, that is, there exists a positive constant d and some n_1, such that for all $n \geq n_1$,

$$T(n) \geq dn$$

By the inductive hypothesis,

$$\begin{aligned} T(n) &\geq 2\left(d\frac{n}{2}\right) + 1 \\ &= dn + 1 \\ &\geq dn \end{aligned}$$

□

Problem 47. *Prove by induction on* n *that the solution to*

$$\begin{aligned} T(n) &= 2T(n-1) + n \\ T(1) &= \Theta(1) \end{aligned} \tag{3.45}$$

is $T(n) = \Theta(2^n)$.

Solution:
We prove separately that $T(n) = O(n)$ and $T(n) = \Omega(n)$.

Part I: Proof that $T(n) = O(n)$. For didactic purposes we will first make several unsuccessful attempts.

Part I, try 1: Assume there exists a positive constant c such that for all large enough n,

$$T(n) \leq c2^n$$

By the inductive hypothesis,

$$\begin{aligned} T(n) &\leq 2c2^{n-1} + n \\ &= c2^n + n \\ &\not\leq c2^n \text{ for any choice of positive } c \end{aligned}$$

Our proof failed so let us strenghten the claim.

Part I, try 2: Assume there exist positive constants b and c such that for all large enough n,

$$T(n) \leq c2^n - b$$

By the inductive hypothesis,

$$\begin{aligned} T(n) &\leq 2(c2^{n-1} - b) + n \\ &= c2^n - 2b + n \\ &\not\leq c2^n - b \text{ for any choice of positive } c \end{aligned}$$

The proof failed once again so let us try another strenghtening of the claim.

Part I, try 3: Assume there exist positive constants b and c such that for all large enough n,

$$T(n) \leq c2^{n-b}$$

By the inductive hypothesis,

$$\begin{aligned} T(n) &\leq 2(c2^{n-b-1}) + n \\ &= c2^{n-b} + n \\ &\not\leq c2^{n-b} \text{ for any choice of positive } c \end{aligned}$$

Yet another failure and we try yet another strenghtening of the claim.

Part I, try 4: Assume there exists a positive constant c such that for all large enough n,

$$T(n) \leq c2^n - n$$

By the inductive hypothesis,

$$\begin{aligned} T(n) &\leq 2(c2^{n-1} - n) + n \\ &= c2^n - n \\ &\leq c2^n - n \text{ for any choice of positive } c \end{aligned}$$

Sucess! At last we have managed to formulate a provable hypothesis.

Part II: Proof that $T(n) = \Omega(n)$, that is, there exists a positive constant d such that for all sufficiently large n,

$$T(n) \geq d2^n$$

By the inductive hypothesis,

$$\begin{aligned} T(n) &\geq 2(d2^{n-1}) + n \\ &= d2^n + n \\ &\geq d2^n \end{aligned}$$

Success! Again we see that the strengthening of the claim is required only in one direction of the proof. □

The next three problems have the iterator

$$n \to \sqrt{n}$$

According to the table on page 37, that number of times this iterator is executed before n becomes some fixed constant is $\Theta(\lg \lg n)$. Note, however, that unless n is integer, this constant cannot be 1 because for real n, it is the case that $n > 1$ after any iteration. Therefore "$T(1) = \Theta(1)$" cannot be the initial condition if n is real. One way out of that is to change the initial conditions to

$$T(n) = \Theta(1) \text{ for } 2 \leq n \leq 4$$

Problem 48. *Solve*

$$T(n) = 2T(\sqrt{n}) + 1 \tag{3.46}$$

Solution:
Substitute n by 2^{2^m}, *i.e.* $m = \lg \lg n$ and $2^m = \lg n$. Then (3.46) becomes

$$T\left(2^{2^m}\right) = 2T\left(2^{\frac{2^m}{2}}\right) + 1$$

which is

$$T\left(2^{2^m}\right) = 2T\left(2^{2^{m-1}}\right) + 1 \tag{3.47}$$

Further substitute $T\left(2^{2^m}\right)$ by $S(m)$ and (3.47) becomes

$$S(m) = 2S(m-1) + 1 \tag{3.48}$$

But we know the solution to that recurrence. According to Problem 39, $S(m) = \Theta(2^m)$. Let us go back now to the original n and $T(n)$.

$$S(m) = \Theta(2^m) \;\Leftrightarrow\; T\left(2^{2^m}\right) = \Theta(\lg n) \;\Leftrightarrow\; T(n) = \Theta(\lg n)$$

□

Problem 49. *Solve*

$$T(n) = 2T(\sqrt{n}) + \lg n \tag{3.49}$$

Solution:
Substitute n by 2^m, *i.e.* $m = \lg n$ and $m = \lg n$. Then (3.49) becomes

$$T(2^m) = 2T\left(2^{\frac{m}{2}}\right) + m \tag{3.50}$$

Further substitute $T\left(2^{2^m}\right)$ by $S(m)$ and (3.50) becomes

$$S(m) = 2S\left(\frac{m}{2}\right) + m \tag{3.51}$$

Consider Problem 41 and Problem 42. They have solve the same recurrence, differing from (3.51) only in the way the division is rounded to integer. In Problem 41 the iterator is

$$n \to \left\lfloor \frac{n}{2} \right\rfloor$$

and in Problem 42 the iterator is

$$n \to \left\lceil \frac{n}{2} \right\rceil$$

Both Problem 41 and Problem 42 have $\Theta(n \lg n)$ solutions. We conclude the solution of (3.51) is $S(m) = \Theta(m \lg m)$, which is equivalent to $T(n) = \Theta(\lg n \lg\lg n)$. □

Problem 50. *Solve*

$$T(n) = \sqrt{n}T(\sqrt{n}) + n \tag{3.52}$$

Solution:
Let us unfold the recurrence:

$$\begin{aligned}
T(n) &= n + n^{\frac{1}{2}}T\left(n^{\frac{1}{2}}\right) && (3.53)\\
&= n + n^{\frac{1}{2}}\left(n^{\frac{1}{2}} + n^{\frac{1}{4}}T\left(n^{\frac{1}{4}}\right)\right) && (3.54)\\
&= 2n + n^{\frac{3}{4}}T\left(n^{\frac{1}{4}}\right) && (3.55)\\
&= 2n + n^{\frac{3}{4}}\left(n^{\frac{1}{8}} + T\left(n^{\frac{1}{8}}\right)\right) && (3.56)\\
&= 3n + n^{\frac{7}{8}}T\left(n^{\frac{1}{8}}\right) && (3.57)\\
&\dots && (3.58)\\
&= in + n^{\left(1-\frac{1}{2^i}\right)}T\left(n^{\frac{1}{2^i}}\right) && (3.59)
\end{aligned}$$

As we already said, the maximum value of i, call it $i_{\max}$, is $i_{\max} = \lg\lg n$. But then $2^{i_{\max}} = \lg n$, therefore

$$n^{\left(1-\frac{1}{2^{i_{\max}}}\right)} = \frac{n}{n^{\frac{1}{2^{i_{\max}}}}} = \frac{n}{n^{\frac{1}{\lg n}}} = \frac{n}{2} \qquad \text{the derivation of } n^{\frac{1}{\lg n}} = 2 \text{ is on page 15}$$

So, for $i = i_{\max}$,

$$T(n) = (\lg\lg n)n + \frac{n}{2}T(c) \qquad c \text{ is some number such that } 2 \le c \le 4$$

But $T(c)$ is a constant, therefore $T(n) = \Theta(n \lg \lg n)$.

Let us prove the same result by induction.
Part 1: Prove that $T(n) = O(n \lg \lg n)$, that is, there exists a positive constant c such that for all sufficiently large n,

$$T(n) \leq cn \lg \lg n \tag{3.60}$$

Our inductive hypothesis then is

$$T(\sqrt{n}) \leq c\sqrt{n} \lg \lg \sqrt{n} \tag{3.61}$$

We know by the definition of the problem that

$$T(n) = \sqrt{n}T(\sqrt{n}) + n \tag{3.62}$$

Apply (3.61) to (3.62) to get

$$\begin{aligned}
T(n) &\leq \sqrt{n}(c\sqrt{n} \lg \lg \sqrt{n}) + n \\
&= cn \lg \lg \sqrt{n} + n \\
&= cn \lg \left(\frac{1}{2} \lg n\right) + n \\
&= cn \lg \left(\frac{\lg n}{2}\right) + n \\
&= cn(\lg \lg n - 1) + n \\
&= cn \lg \lg n - cn + n \\
&\leq cn \lg \lg n \qquad \text{provided that } -cn + n \leq 0 \Leftrightarrow c \geq 1
\end{aligned}$$

Part 2: Prove that $T(n) = \Omega(n \lg \lg n)$, that is, there exists a positive constant d such that for all sufficiently large n,

$$T(n) \geq dn \lg \lg n \tag{3.63}$$

Our inductive hypothesis then is

$$T(\sqrt{n}) \geq d\sqrt{n} \lg \lg \sqrt{n} \tag{3.64}$$

We know by the definition of the problem that

$$T(n) = \sqrt{n}T(\sqrt{n}) + n \tag{3.65}$$

Apply (3.64) to (3.65) to get

$$\begin{aligned}
T(n) &\geq \sqrt{n}(d\sqrt{n} \lg \lg \sqrt{n}) + n \\
&= dn \lg \lg \sqrt{n} + n \\
&= dn \lg \left(\frac{1}{2} \lg n\right) + n \\
&= dn \lg \left(\frac{\lg n}{2}\right) + n \\
&= dn(\lg \lg n - 1) + n \\
&= dn \lg \lg n - dn + n \\
&\geq dn \lg \lg n \qquad \text{provided that } -dn + n \geq 0 \Leftrightarrow d \leq 1
\end{aligned}$$

□

Problem 51. *Solve by unfolding*

$$T(n) = T(n-2) + 2\lg n \tag{3.66}$$

Solution:
Let us unfold the recurrence:

$$\begin{aligned} T(n) &= T(n-2) + 2\lg n \\ &= T(n-4) + 2\lg(n-2) + 2\lg n \\ &= T(n-6) + 2\lg(n-4) + 2\lg(n-2) + 2\lg n \\ &= \dots \\ &= T(c) + \dots + 2\lg(n-4) + 2\lg(n-2) + 2\lg n \end{aligned} \tag{3.67}$$

where c is either 1 or 2[†].

Case I: n is odd. Then $c = 1$ and (3.67) is:

$$2\lg n + 2\lg(n-2) + 2\lg(n-4) + \dots + 2\lg 3 + T(1) \tag{3.68}$$

We approximate $T(1)$ with $0 = \lg 1$, which does not alter the asymptotic growth rate of (3.68), and thus (3.68) becomes:

$$\begin{aligned} &\lg n^2 + \lg(n-2)^2 + \lg(n-4)^2 + \dots + \lg 3^2 + \lg 1 = \\ &\lg\left(n^2(n-2)^2(n-4)^2 \dots 3^2.1\right) = \\ &\lg\Big(\underbrace{n.n(n-2)(n-2)(n-4)(n-4)\dots 5.5.3.3.1}_{n \text{ factors}}\Big) = T(n) \end{aligned} \tag{3.69}$$

Define

$$\begin{aligned} X(n) &= \lg\Big(\underbrace{n(n-1)(n-2)(n-3)\dots 3.2.1}_{n \text{ factors}}\Big) = \lg n! \\ Y(n) &= \lg\Big(\underbrace{(n+1)n(n-1)(n-2)\dots 4.3.2}_{n \text{ factors}}\Big) = \lg(n+1)! \end{aligned}$$

and note that

$$X(n) \le T(n) \le Y(n) \tag{3.70}$$

because of the following inequalities between the corresponding factors inside the logarithms

$$\begin{array}{lccccccccc} X(n) = \lg\big(& n & n-1 & n-2 & n-3 & \cdots & 3 & 2 & 1 & \big) \\ & /\!/\!\backslash & /\!/\!\backslash & /\!/\!\backslash & /\!/\!\backslash & & /\!/\!\backslash & /\!/\!\backslash & /\!/\!\backslash & \\ T(n) = \lg\big(& n & n & n-2 & n-2 & \cdots & 3 & 3 & 1 & \big) \\ & /\!/\!\backslash & /\!/\!\backslash & /\!/\!\backslash & /\!/\!\backslash & & /\!/\!\backslash & /\!/\!\backslash & /\!/\!\backslash & \\ Y(n) = \lg\big(& n+1 & n & n-1 & n-2 & \cdots & 4 & 3 & 2 & \big) \end{array}$$

[†]The initial conditions that define $T(1)$ and $T(2)$ are omitted.

However, $X(n) = \Theta(n \lg n)$ and $Y(n) = \Theta((n+1)\lg(n+1)) = \Theta(n \lg n)$ by (1.42). Having in mind that and (3.70), $T(n) = \Theta(n \lg n)$ follows immediately.

Case II: n is even. Then $c = 2$ and (3.67) is:

$$2\lg n + 2\lg(n-2) + 2\lg(n-4) + \ldots + 2\lg 4 + T(2) \tag{3.71}$$

We approximate $T(2)$ with $1 = \lg 2$, which does not alter the asymptotic growth rate of (3.68), and thus (3.68) becomes:

$$\begin{aligned}
&\lg n^2 + \lg(n-2)^2 + \lg(n-4)^2 + \ldots + \lg 4^2 + \lg 2 = \\
&\lg\left(n^2(n-2)^2(n-4)^2 \ldots 4^2.2\right) = \\
&\lg\Big(\underbrace{n.n(n-2)(n-2)(n-4)(n-4)\ldots 6.6.4.4.2}_{n-1 \text{ factors}}\Big) = T(n)
\end{aligned} \tag{3.72}$$

Define

$$X(n) = \lg\Big(\underbrace{n(n-1)(n-2)(n-3)\ldots 4.3.2}_{n-1 \text{ factors}}\Big) = \lg n!$$

$$Y(n) = \lg\Big(\underbrace{(n+1)n(n-1)(n-2)\ldots 5.4.3}_{n-1 \text{factors}}\Big) = \lg\frac{(n+1)!}{2} = \lg(n+1)! - 1$$

and note that

$$X(n) \le T(n) \le Y(n) \tag{3.73}$$

because of the following inequalities between the corresponding factors inside the logarithms

$$\begin{array}{lccccccccc}
X(n) = \lg\big(& n & n-1 & n-2 & n-3 & \cdots & 4 & 3 & 2 & \big) \\
 & /\!/\!\backslash & /\!/\!\backslash & /\!/\!\backslash & /\!/\!\backslash & & /\!/\!\backslash & /\!/\!\backslash & /\!/\!\backslash & \\
T(n) = \lg\big(& n & n & n-2 & n-2 & \cdots & 4 & 4 & 2 & \big) \\
 & /\!/\!\backslash & /\!/\!\backslash & /\!/\!\backslash & /\!/\!\backslash & & /\!/\!\backslash & /\!/\!\backslash & /\!/\!\backslash & \\
Y(n) = \lg\big(& n+1 & n & n-1 & n-2 & \cdots & 5 & 4 & 3 & \big)
\end{array}$$

However, $X(n) = \Theta(n \lg n)$ and $Y(n) = \Theta((n+1)\lg(n+1)) = \Theta(n \lg n)$ by (1.42). Having in mind that and (3.70), $T(n) = \Theta(n \lg n)$ follows immediately. □

Problem 52. *Solve by induction*

$$T(n) = T(n-2) + 2\lg n \tag{3.74}$$

Solution:
We use Problem 51 to guess the solution $T(n) = \Theta(n \lg n)$.

Part I: Proof that $T(n) = O(n \lg n)$, that is, there exists a positive constant c such that for all sufficiently large n,

$$T(n) \le cn \lg n \tag{3.75}$$

The following inequalities hold

$$\begin{aligned}
T(n) &\leq c(n-2)\lg(n-2)+2\lg n \qquad \text{from the induction hypothesis}\\
&\leq c(n-2)\lg n+2\lg n\\
&= cn\lg n-2c\lg n+2\lg n\\
&\leq cn\lg n \qquad \text{provided that } -2c\lg n+2\lg n\leq 0 \Leftrightarrow c\geq 1
\end{aligned}$$

Part II: Proof that $T(n)=\Omega(n\lg n)$, that is, there exists a positive constant d such that for all sufficiently large n,

$$T(n)\geq dn\lg n \tag{3.76}$$

It is the case that

$$\begin{aligned}
T(n) &\geq d(n-2)\lg(n-2)+2\lg n \qquad \text{from the induction hypothesis}\\
&= (dn-2d)\lg(n-2)+2\lg n\\
&= dn\lg(n-2)+2(\lg n-d\lg(n-2))
\end{aligned} \tag{3.77}$$

Having in mind (3.76) and (3.77), our goal is to show that

$$\begin{aligned}
dn\lg(n-2)+2(\lg n-d\lg(n-2)) &\geq dn\lg n \Leftrightarrow\\
dn\lg(n-2)-dn\lg n+2(\lg n-d\lg(n-2)) &\geq 0 \Leftrightarrow\\
\underbrace{d\lg\left(\frac{n-2}{n}\right)^n}_{A}+\underbrace{2\lg\frac{n}{(n-2)^d}}_{B} &\geq 0
\end{aligned} \tag{3.78}$$

Let us first evaluate A when n grows infinitely:

$$\lim_{n\to\infty} d\lg\left(\frac{n-2}{n}\right)^n = d\lim_{n\to\infty}\lg\left(1+\frac{-2}{n}\right)^n = d\lg\lim_{n\to\infty}\left(1+\frac{-2}{n}\right)^n = d\lg e^{-2} = -2d\lg e$$

Now consider B when n grows infinitely:

$$\lim_{n\to\infty} 2\lg\frac{n}{(n-2)^d} = 2\lg\lim_{n\to\infty}\frac{n}{(n-2)^d} \tag{3.79}$$

Note that for any d such that $0<d<1$, (3.79) is $+\infty$. For instance, for $d=\frac{1}{2}$, (3.79) becomes

$$2\lg\lim_{n\to\infty}\left(n^{\frac{1}{2}}\frac{n^{\frac{1}{2}}}{(n-2)^{\frac{1}{2}}}\right) =$$

$$2\lg\lim_{n\to\infty}\left(n^{\frac{1}{2}}\left(\frac{n}{n-2}\right)^{\frac{1}{2}}\right) =$$

$$2\lg\left(\left(\lim_{n\to\infty}n^{\frac{1}{2}}\right)\underbrace{\left(\lim_{n\to\infty}\left(\frac{1}{1-\frac{2}{n}}\right)^{\frac{1}{2}}\right)}_{1}\right) = +\infty$$

It follows inequality (3.78) is true for any choice of d such that $0 < d < 1$, say, $d = \frac{1}{2}$, because A by absolute value is limited by a constant, and B grows infinitely. And that concludes the proof of (3.75). □

⁘ NB ⁘ The proof by induction in **Part II** of the solution to Problem 51 is tricky. Consider (3.77):

$$dn \lg(n-2) + 2(\lg n - d \lg(n-2))$$

Typically, we deal with logarithms of additions or differences by approximating the additions or differences with multiplications or fractions. But notice that if we approximate $n-2$ inside the above logarithms with *any* fraction $\frac{n}{\alpha}$, for any positive constant α, we cannot accomplish the proof. Here is what happens when we substitute $n-2$ with $\frac{n}{\alpha}$ in the logarithm on the left:

$$dn \lg \frac{n}{\alpha} + 2(\lg n - d \lg(n-2)) = dn \lg n - d\alpha n + 2(\lg n - d \lg(n-2))$$

To accomplish the proof, we have to show the latter is greater than or equal to $dn \lg n$; and to show that, we have to show that the term $-d\alpha n + 2(\lg n - d \lg(n-2))$ is positive. But that is not true! Both d and α are positive constants, so $-d\alpha n$ is necessarily negative for positive values of n. And the asymptotic behaviour of $-d\alpha n + 2(\lg n - d \lg(n-2))$ is determined by $-d\alpha n$ because the linear function dominates the logarithmic function for all sufficiently large n. Therefore, we need a more sophisticated technique, based on analysis.

Problem 53. *Solve by unfolding*

$$T(n) = T(n-1) + \lg n$$

Solution:

$$\begin{aligned}
T(n) &= T(n-1) + \lg n \\
&= T(n-2) + \lg(n-1) + \lg n \\
&= T(n-3) + \lg(n-2) + \lg(n-1) + \lg n \\
&\dots \\
&= \underbrace{T(1)}_{\Theta(1)} + \lg 2 + \lg 3 + \ldots + \lg(n-2) + \lg(n-1) + \lg n \\
&= \Theta(1) + \lg(2.3 \ldots (n-2)(n-1)n) \\
&= \Theta(1) + \lg n! \\
&= \Theta(1) + \Theta(n \lg n) \qquad \text{by (1.42)} \\
&= \Theta(n \lg n)
\end{aligned}$$

□

Problem 54. *Solve by unfolding*

$$T(n) = 3T\left(\left\lfloor \frac{n}{4} \right\rfloor\right) + n \tag{3.80}$$

Solution:

$$
\begin{aligned}
T(n) &= n + 3T\left(\left\lfloor \frac{n}{4} \right\rfloor\right) \\
&= n + 3\left(\left\lfloor \frac{n}{4} \right\rfloor + 3T\left(\left\lfloor \frac{\left\lfloor \frac{n}{4} \right\rfloor}{4} \right\rfloor\right)\right) \\
&= n + 3\left(\left\lfloor \frac{n}{4} \right\rfloor + 3T\left(\left\lfloor \frac{n}{16} \right\rfloor\right)\right) \qquad \text{because } \left\lfloor \frac{\left\lfloor \frac{n}{4} \right\rfloor}{4} \right\rfloor = \left\lfloor \frac{n}{16} \right\rfloor \\
&= n + 3\left\lfloor \frac{n}{4} \right\rfloor + 9T\left(\left\lfloor \frac{n}{16} \right\rfloor\right) \\
&= n + 3\left\lfloor \frac{n}{4} \right\rfloor + 9\left\lfloor \frac{n}{16} \right\rfloor + 27T\left(\left\lfloor \frac{n}{64} \right\rfloor\right) \\
&\dots \\
&= \underbrace{3^0\left\lfloor \frac{n}{4^0} \right\rfloor + 3^1\left\lfloor \frac{n}{4^1} \right\rfloor + 3^2\left\lfloor \frac{n}{4^2} \right\rfloor + \ldots + 3^{i-1}\left\lfloor \frac{n}{4^{i-1}} \right\rfloor}_{\text{main part } P(n)} + \underbrace{3^i T\left(\left\lfloor \frac{n}{4^i} \right\rfloor\right)}_{\text{remainder}} \qquad (3.81)
\end{aligned}
$$

The maximum value for i, let us call it i_{max}, is achieved when $\left\lfloor \frac{n}{4^i} \right\rfloor$ becomes 1. It follows $i_{max} = \lfloor \log_4 n \rfloor$. Let us estimate the main part and the remainder of (3.81) for $i = i_{max}$.

- To estimate the main part, define

$$
\begin{aligned}
X(n) &= 3^0\left(\frac{n}{4^0}\right) + 3^1\left(\frac{n}{4^1}\right) + 3^2\left(\frac{n}{4^2}\right) + \ldots + 3^{i_{max}-1}\left(\frac{n}{4^{i_{max}-1}}\right) \\
Y(n) &= 3^0\left(\frac{n}{4^0} + 1\right) + 3^1\left(\frac{n}{4^1} + 1\right) + 3^2\left(\frac{n}{4^2} + 1\right) + \ldots + 3^{i_{max}-1}\left(\frac{n}{4^{i_{max}-1}} + 1\right)
\end{aligned}
$$

  Clearly, $X(n) \le P(n) \le Y(n)$. But

$$
\begin{aligned}
X(n) &= n\sum_{j=0}^{i_{max}-1}\left(\frac{3}{4}\right)^j \\
&\le n\sum_{j=0}^{\infty}\left(\frac{3}{4}\right)^j \\
&= n\frac{1}{1-\frac{3}{4}} \\
&= 4n \\
&= \Theta(n)
\end{aligned}
$$

and

$$\begin{aligned} Y(n) &= n \sum_{j=0}^{i_{\max}-1} \left(\frac{3}{4}\right)^j + \sum_{j=0}^{i_{\max}-1} 3^j \\ &\leq 4n + \sum_{j=0}^{i_{\max}-1} 3^j \\ &= 4n + \Theta\left(3^{i_{\max}-1}\right) \qquad \text{by Problem 28} \\ &= 4n + \Theta\left(3^{\log_4 n}\right) \qquad i_{\max} = \lfloor \log_4 n \rfloor \text{ and } 3^{\lfloor \log_4 n \rfloor} = \Theta(3^{\log_4 n}) \\ &= \Theta(n) \end{aligned}$$

Then it has to be the case that $X(n) = \Theta(n)$.

- To estimate the remainder, consider the two factors in it:

$$3^{i_{\max}} = 3^{\lfloor \log_4 n \rfloor} = \Theta(3^{\log_4 n}) = \Theta(n^{\log_3 4})$$
$$T\left(\left\lfloor \frac{n}{4^{i_{\max}}} \right\rfloor\right) = T(1) = \Theta(1)$$

It follows the remainder is $\Theta(3^{\log_4 n}) = o(n)$.

Therefore, $T(n) = \Theta(n) + o(n) = \Theta(n)$. □

Problem 55. *Solve*

$$T(n) = 2T\left(\frac{n}{2}\right) + n^2$$

by the method of the recursion tree.

Solution:
The recursion tree is shown on Figure 3.6. The solution is the sum

$$n^2 + \frac{n^2}{2} + \frac{n^2}{4} + \frac{n^2}{8} + \ldots \leq n^2 \sum_{i=0}^{\infty} \frac{1}{2^i} = 2n^2$$

It follows $T(n) = \Theta(n^2)$. □

Problem 56. *Solve*

$$T(n) = T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + n$$

by the method of the recursion tree.

Solution:
The recursion tree is shown on Figure 3.7. This time the tree is not complete so we do not write the levels on the left side in terms of n (as we did on Figure 3.6). Rather, the level of each node is the distance between it and the root. Thus the equidistant with respect to the

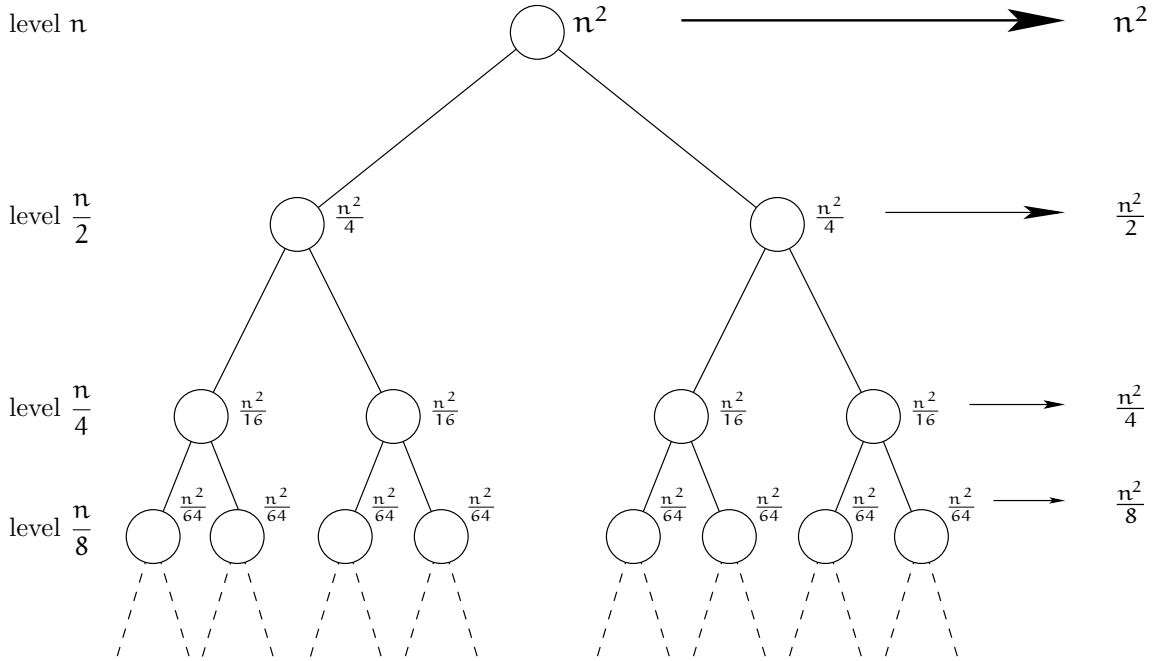

Figure 3.6: The recursion tree of $T(n) = 2T\left(\frac{n}{2}\right) + n^2$.

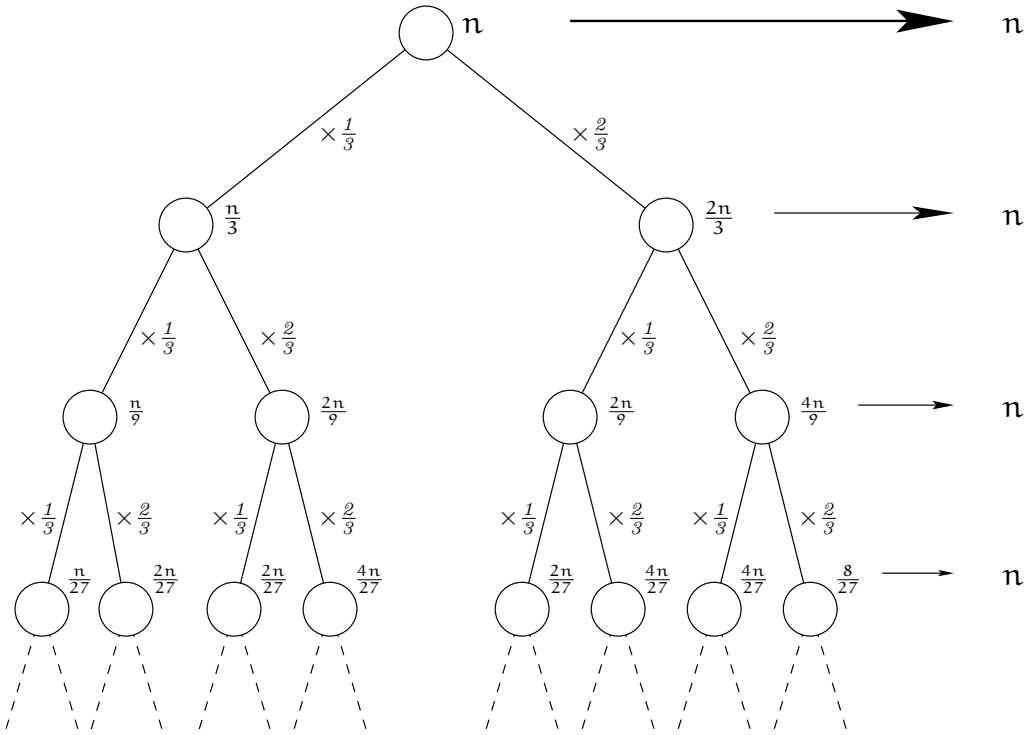

Figure 3.7: The recursion tree of $T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + n$.

root nodes are at the same level. Think of the tree as an ordered tree. That is, if a node has any children we distinguish between the left and the right child. The value of the left child is the value of the parent multiplied by $\frac{1}{3}$ and the value of the right child is the value of the parent multiplied by $\frac{2}{3}$. It is trivial to prove by induction that for each level such that all the nodes at this level exist, the sum of the values at that level is n. However, we cannot obtain the answer immediately through mulitplying n by the height because the tree is not balanced. The maximum distance between the root and any leaf is achieved along the rightmost path (starting at the root, always take the right choice; see Figure 3.7) and the minimum distance, by the leftmost path. The length of the leftmost path is determined by the iterator

$$n \to \frac{n}{3}$$

which is executed $\Theta(\log_3 n)$ times before reaching any fixed in advance constant. The length of the rightmost path is determined by the iterator

$$n \to \frac{2n}{3} = \frac{n}{\frac{3}{2}}$$

which is executed $\Theta\left(\log_{\frac{3}{2}} n\right)$ times before reaching any fixed in advance constant.

Let $\mathcal{T}$ be the recursion tree. Construct two balanced trees $\mathcal{T}_1$ and $\mathcal{T}_2$ such that the height of $\mathcal{T}_1$ is $\Theta(\log_3 n)$ and the height of $\mathcal{T}_2$ is $\Theta\left(\log_{\frac{3}{2}} n\right)$. Suppose that each level in $\mathcal{T}_1$ and $\mathcal{T}_2$ is associated with some value n – it does not matter for what reason, just assume each level "costs" n. Let $S_i(n)$ be the sum of those costs in $\mathcal{T}_i$ over all levels for $i = 1, 2$. Clearly,

$$S_1(n) = n \times \Theta(\log_3 n) = \Theta(n \log_3 n) = \Theta(n \lg n)$$
$$S_2(n) = n \times \Theta\left(\log_{\frac{3}{2}} n\right) = \Theta\left(n \log_{\frac{3}{2}} n\right) = \Theta(n \lg n)$$

To conlude the solution, note that $S_1(n) \leq T(n) \leq S_2(n)$ because $\mathcal{T}_1$ can be considered a subtree of $\mathcal{T}$ and $\mathcal{T}$ can be considered a subtree of $\mathcal{T}_2$. Then $T(n) = \Theta(n \lg n)$. □

Problem 57. *Solve by unfolding*

$$T(n) = T(n - a) + T(a) + n \qquad a = const,\ a \geq 1$$

Solution:
We assume a is integer[†] and the initial conditions are

$$T(1) = \Theta(1)$$
$$T(2) = \Theta(1)$$
$$\dots$$
$$T(a) = \Theta(1)$$

[†]It is not essential to postulate a is integer. The problems makes sense even if a is just a positive real. If that is the case the initial conditions have to be changed to cover some interval with length a, *e.g.* $T(i) = \text{const.}$ if $i \in (0, a]$.

Let us unfold the recurrence.

$$\begin{aligned}
T(n) &= T(n-a) + T(a) + n \\
&= (T(n-2a) + T(a) + n - a) + n \\
&= T(n-2a) + 2T(a) + 2n - a \\
&= (T(n-3a) + T(a) + n - 2a) + 2T(a) + 2n - a \\
&= T(n-3a) + 3T(a) + 3n - 3a \\
&= (T(n-4a) + T(a) + n - 4a) + 3T(a) + 3n - 3a \\
&= T(n-4a) + 4T(a) + 4n - 6a \\
&= (T(n-5a) + T(a) + n - 4a) + 4T(a) + 4n - 6a \\
&= T(n-5a) + 5T(a) + 5n - 10a \\
&\dots \\
&= T(n-ia) + iT(a) + in - \frac{1}{2}i(i-1)a
\end{aligned} \tag{3.82}$$

Let the maximum value i takes be i_{max}. Consider the iterator

$$n \to n - a$$

It maps every $n > a$, $n \in \mathbb{N}$, to a unique number from $\{1, 2, \dots, a\}$. Let that number be called k. So i_{max} is the number of times the iterator is executed until the variable becomes k. If $n \bmod a \neq 0$ then k is $n \bmod a$, otherwise k is a[†]. It follows that

$$i_{max} = \begin{cases} \left\lfloor \frac{n}{a} \right\rfloor, & \text{if } n \bmod a \neq 0 \\ \frac{n}{a} - 1, & \text{else} \end{cases}$$

That is equivalent to

$$i_{max} = \left\lceil \frac{n}{a} \right\rceil - 1$$

Subsituting i with $\left\lceil \frac{n}{a} \right\rceil - 1$ in (3.82), we get

$$T(k) + \left(\left\lceil \frac{n}{a} \right\rceil - 1 \right) T(a) + \left(\left\lceil \frac{n}{a} \right\rceil - 1 \right) n - \frac{1}{2} \left(\left\lceil \frac{n}{a} \right\rceil - 1 \right) \left(\left\lceil \frac{n}{a} \right\rceil - 1 - 1 \right) a \tag{3.83}$$

The growth rate of (3.83) is determined by

$$n \left\lceil \frac{n}{a} \right\rceil - \frac{1}{2} \left\lceil \frac{n}{a} \right\rceil \left\lceil \frac{n}{a} \right\rceil = \Theta(n^2)$$

It follows $T(n) = \Theta(n^2)$. □

Problem 58. *Solve*

$$T(n) = T(\alpha n) + T((1-\alpha)n) + n, \qquad \alpha = const.,\ 0 < \alpha < 1 \tag{3.84}$$

by the method of the recursion tree.

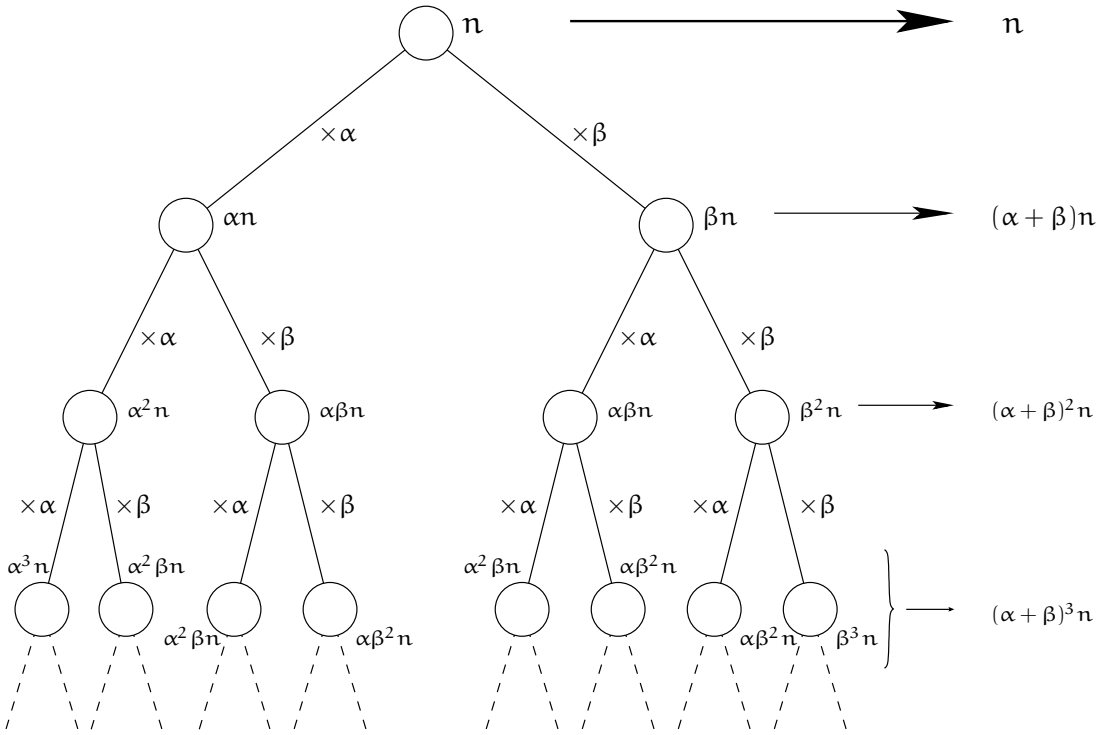

Figure 3.8: The recursion tree of $T(n) = T(\alpha n) + T(\beta n) + n$ where $0 < \alpha, \beta < 1$ and $\alpha + \beta = 1$.

Solution:
Define that $1 - \alpha = \beta$. Obviously, $0 < \beta < 1$ and (3.84) becomes

$$T(n) = T(\alpha n) + T(\beta n) + n \tag{3.85}$$

The recursion tree of (3.85) is shown on Figure 3.8. The solution is completely analogous to the solution of Problem 56. The level of each node is the distance between it and the root. The sum of the costs at every level such that all nodes at that levels exist, is n. More precisely, at level i the sum is $(\alpha + \beta)^i n = n$. The tree is not complete. Assume without loss of generality that $\alpha \leq \beta$ and think of the tree as an ordered tree. The shortest path from the root to any leaf is the leftmost one, *i.e.* "follow the alphas", and the longest path is the rightmost one. The length of the shortest path is $\log_{\left(\frac{1}{\alpha}\right)} n$ and of the longest path, $\log_{\left(\frac{1}{\beta}\right)} n$. We prove that $T(n) = \Theta(n \lg n)$ just as in Problem 56 by considering two other trees, one that is a subgraph of the current one and one that isa supergraph of the current one. Since the first of then has sum of the costs $n \times \Theta\left(\log_{\left(\frac{1}{\alpha}\right)} n\right) = \Theta(n \lg n)$ and the second one, $n \times \Theta\left(\log_{\left(\frac{1}{\beta}\right)} n\right) = \Theta(n \lg n)$, it follows $T(n) = \Theta(n \lg n)$. □

Problem 59. *Solve*

$$T(n) = T(n-1) + \frac{1}{n} \tag{3.86}$$

[†]Not $n \bmod a$, which is 0.

Solution:
We solve the recurrence by unfolding. Before we commence the unfolding check the definition of the harmonic series, the partial sum H_n of the harmonic series, and its order of growth $\Theta(\lg n)$ on page 91.

$$\begin{aligned}
T(n) &= T(n-1) + \frac{1}{n} \\
&= T(n-2) + \frac{1}{n-1} + \frac{1}{n} \\
&= T(n-3) + \frac{1}{n-2} + \frac{1}{n-1} + \frac{1}{n} \\
&\dots \\
&= T(1) + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n-2} + \frac{1}{n-1} + \frac{1}{n} \\
&= T(1) - 1 + \underbrace{1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n-2} + \frac{1}{n-1} + \frac{1}{n}}_{H_n} \\
&= O(1) + H_n \\
&= O(1) + \Theta(\lg n) \\
&= \Theta(\lg n)
\end{aligned}$$

□

Problem 60. *Solve by unfolding*

$$T(n) = \frac{n}{n+1} T(n-1) + 1$$

Solution:

$$\begin{aligned}
T(n) &= \frac{n}{n+1} T(n-1) + 1 \\
&= \frac{n}{n+1} \left(\frac{n-1}{n} T(n-2) + 1 \right) + 1 \\
&= \frac{n-1}{n+1} T(n-2) + \frac{n}{n+1} + 1 \\
&= \frac{n-1}{n+1} \left(\frac{n-2}{n-1} T(n-3) + 1 \right) + \frac{n}{n+1} + 1 \\
&= \frac{n-2}{n+1} T(n-3) + \frac{n-1}{n+1} + \frac{n}{n+1} + 1 \\
&= \frac{n-2}{n+1} \left(\frac{n-3}{n-2} T(n-4) + 1 \right) + \frac{n-1}{n+1} + \frac{n}{n+1} + 1 \\
&= \frac{n-3}{n+1} T(n-4) + \frac{n-2}{n+1} + \frac{n-1}{n+1} + \frac{n}{n+1} + 1 \qquad (3.87)
\end{aligned}$$

If we go on like that down to $T(1)$, (3.87) unfolds into

$$\begin{aligned}
T(n) &= \frac{2}{n+1}T(1) + \frac{3}{n+1} + \frac{4}{n+1} + \ldots + \frac{n-2}{n+1} + \frac{n-1}{n+1} + \frac{n}{n+1} + 1 \\
&= \frac{2}{n+1}T(1) + \frac{3}{n+1} + \frac{4}{n+1} + \ldots + \frac{n-2}{n+1} + \frac{n-1}{n+1} + \frac{n}{n+1} + \frac{n+1}{n+1} \\
&= \frac{2T(1)}{n+1} + \frac{1}{n+1}\sum_{i=3}^{n+1} i \\
&= \frac{2T(1)}{n+1} + \frac{1}{n+1}\left(\left(\sum_{i=1}^{n+1} i\right) - 3\right) \\
&= \frac{2T(1)}{n+1} + \frac{1}{n+1}\left(\frac{(n+1)(n+2)}{2} - 3\right) \\
&= \frac{1}{n+1}\left(4T(1) + (n^2+3n+2) - 6\right) \\
&= \frac{n^2+3n+4T(1)-4}{n+1} \\
&= \underbrace{\frac{n^2}{n+1}}_{\Theta(n)} + \underbrace{\frac{3n}{n+1}}_{\Theta(1)} + \underbrace{\frac{4T(1)-4}{n+1}}_{O(1)} \\
&= \Theta(n)
\end{aligned}$$

So, $T(n) = \Theta(n)$. □

Problem 61. *Solve by unfolding*

$$T(n) = \frac{n}{n+1}T(n-1) + n^2$$

Solution:

$$\begin{aligned}
T(n) &= \frac{n}{n+1}T(n-1) + n^2 \\
&= \frac{n}{n+1}\left(\frac{n-1}{n}T(n-2) + (n-1)^2\right) + n^2 \\
&= \frac{n-1}{n+1}T(n-2) + \frac{n(n-1)^2}{n+1} + n^2 \\
&= \frac{n-1}{n+1}\left(\frac{n-2}{n-1}T(n-3) + (n-2)^2\right) + \frac{n(n-1)^2}{n+1} + n^2 \\
&= \frac{n-2}{n+1}T(n-3) + \frac{(n-1)(n-2)^2}{n+1} + \frac{n(n-1)^2}{n+1} + n^2 \\
&= \frac{n-2}{n+1}\left(\frac{n-3}{n-2}T(n-4) + (n-3)^2\right) + \frac{(n-1)(n-2)^2}{n+1} + \frac{n(n-1)^2}{n+1} + n^2 \\
&= \frac{n-3}{n+1}T(n-4) + \frac{(n-2)(n-3)^2}{n+1} + \frac{(n-1)(n-2)^2}{n+1} + \frac{n(n-1)^2}{n+1} + n^2 \quad (3.88)
\end{aligned}$$

If we go on like that down to $T(1)$, (3.88) unfolds into

$$\begin{aligned}
T(n) &= \frac{2}{n+1}T(1) + \frac{3.2^2}{n+1} + \frac{4.3^2}{n+1} + \dots \\
&\qquad + \frac{(n-2)(n-3)^2}{n+1} + \frac{(n-1)(n-2)^2}{n+1} + \frac{n(n-1)^2}{n+1} + n^2 \\
&= \frac{2}{n+1}T(1) + \frac{3.2^2}{n+1} + \frac{4.3^2}{n+1} + \dots \\
&\qquad + \frac{(n-2)(n-3)^2}{n+1} + \frac{(n-1)(n-2)^2}{n+1} + \frac{n(n-1)^2}{n+1} + \frac{(n+1)n^2}{n+1} \\
&= \frac{2T(1)}{n+1} + \frac{1}{n+1}\sum_{i=3}^{n+1} i(i-1)^2 \\
&= \frac{2T(1)}{n+1} + \frac{1}{n+1}\left(\left(\sum_{i=1}^{n+1} i(i-1)^2\right) - 2\right) \\
&= \frac{2T(1)-2}{n+1} + \frac{1}{n+1}\sum_{i=1}^{n+1} i(i-1)^2 \\
&= \frac{2T(1)-2}{n+1} + \frac{1}{n+1}\sum_{i=1}^{n+1} (i^3 - 2i^2 + i) \\
&= \frac{2T(1)-2}{n+1} + \frac{1}{n+1}\left(\sum_{i=1}^{n+1} i^3 - 2\sum_{i=1}^{n+1} i^2 + \sum_{i=1}^{n+1} i\right) \qquad (3.89)
\end{aligned}$$

Having in mind (4.21), (4.22), and (4.23) on page 92, (3.89) becomes

$$\begin{aligned}
&\frac{2T(1)-2}{n+1} + \frac{1}{n+1}\left(\frac{(n+1)^2(n+2)^2}{4} - 2\frac{(n+1)(n+2)(2n+3)}{6} + \frac{(n+1)(n+2)}{2}\right) \\
&= \underbrace{\frac{2T(1)-2}{n+1}}_{O(1)} + \underbrace{\frac{(n+1)(n+2)^2}{4}}_{\Theta(n^3)} - \underbrace{\frac{(n+2)(2n+3)}{3}}_{\Theta(n^2)} + \underbrace{\frac{n+2}{2}}_{\Theta(n)} \\
&= \Theta(n^3)
\end{aligned}$$

So, $T(n) = \Theta(n^3)$. □

Problem 62. *Solve by unfolding*

$$T(n) = \frac{n}{n+1}T(n-1) + \sqrt{n} \qquad (3.90)$$

where $\sqrt{n}$ *stands for either* $\lfloor\sqrt{n}\rfloor$ *or* $\lceil\sqrt{n}\rceil$.

Solution:

$$
\begin{aligned}
T(n) &= \frac{n}{n+1}T(n-1) + \sqrt{n} \\
&= \frac{n}{n+1}\left(\frac{n-1}{n}T(n-2) + \sqrt{n-1}\right) + \sqrt{n} \\
&= \frac{n-1}{n+1}T(n-2) + \frac{n\sqrt{n-1}}{n+1} + \sqrt{n} \\
&= \frac{n-1}{n+1}\left(\frac{n-2}{n-1}T(n-3) + \sqrt{n-2}\right) + \frac{n\sqrt{n-1}}{n+1} + \sqrt{n} \\
&= \frac{n-2}{n+1}T(n-3) + \frac{(n-1)\sqrt{n-2}}{n+1} + \frac{n\sqrt{n-1}}{n+1} + \sqrt{n} \\
&= \frac{n-2}{n+1}\left(\frac{n-3}{n-2}T(n-4) + \sqrt{n-3}\right) + \frac{(n-1)\sqrt{n-2}}{n+1} + \frac{n\sqrt{n-1}}{n+1} + \sqrt{n} \\
&= \frac{n-3}{n+1}T(n-4) + \frac{(n-2)\sqrt{n-3}}{n+1} + \frac{(n-1)\sqrt{n-2}}{n+1} + \frac{n\sqrt{n-1}}{n+1} + \sqrt{n} \qquad (3.91)
\end{aligned}
$$

If we go on like that down to $T(1)$, (3.91) unfolds into

$$
\begin{aligned}
T(n) &= \frac{2}{n+1}T(1) + \frac{3\sqrt{2}}{n+1} + \frac{4\sqrt{3}}{n+1} + \ldots \\
&\qquad + \frac{(n-2)\sqrt{n-3}}{n+1} + \frac{(n-1)\sqrt{n-2}}{n+1} + \frac{n\sqrt{n-1}}{n+1} + \sqrt{n} \\
&= \frac{2}{n+1}T(1) + \frac{3\sqrt{2}}{n+1} + \frac{4\sqrt{3}}{n+1} + \ldots \\
&\qquad + \frac{(n-2)\sqrt{n-3}}{n+1} + \frac{(n-1)\sqrt{n-2}}{n+1} + \frac{n\sqrt{n-1}}{n+1} + \frac{(n+1)\sqrt{n}}{n+1} \\
&= \frac{2T(1)}{n+1} + \frac{1}{n+1}\sum_{i=2}^{n}(i+1)\sqrt{i} \\
&= \frac{2T(1)}{n+1} + \frac{1}{n+1}\left(\left(\sum_{i=1}^{n}(i+1)\sqrt{i}\right) - \sqrt{2}\right) \\
&= \frac{2T(1)-\sqrt{2}}{n+1} + \frac{1}{n+1}\sum_{i=1}^{n}(i+1)\sqrt{i} \\
&= \frac{2T(1)-\sqrt{2}}{n+1} + \frac{1}{n+1}\sum_{i=1}^{n}(i\sqrt{i} + \sqrt{i}) \\
&= \frac{2T(1)-\sqrt{2}}{n+1} + \frac{1}{n+1}\left(\sum_{i=1}^{n} i\sqrt{i} + \sum_{i=1}^{n}\sqrt{i}\right) \qquad (3.92)
\end{aligned}
$$

But we know that

$$\sum_{i=1}^{n} \left\lfloor \sqrt{i} \right\rfloor = \Theta\left(n^{\frac{3}{2}}\right) \qquad \text{by (4.5) on page 83.}$$

$$\sum_{i=1}^{n} \left\lceil \sqrt{i} \right\rceil = \Theta\left(n^{\frac{3}{2}}\right) \qquad \text{by (4.7) on page 85.}$$

$$\sum_{i=1}^{n} i \left\lfloor \sqrt{i} \right\rfloor = \Theta\left(n^{\frac{5}{2}}\right) \qquad \text{by (4.10) on page 87.}$$

$$\sum_{i=1}^{n} i \left\lceil \sqrt{i} \right\rceil = \Theta\left(n^{\frac{5}{2}}\right) \qquad \text{by (4.14) on page 90.}$$

Therefore, regardless of whether "$\sqrt{n}$" in (3.90) stands for $\lfloor\sqrt{n}\rfloor$ or $\lceil\sqrt{n}\rceil$,

$$\begin{aligned} T(n) &= \frac{2T(1)-\sqrt{2}}{n+1} + \frac{1}{n+1}\left(\Theta\left(n^{\frac{5}{2}}\right) + \Theta\left(n^{\frac{5}{2}}\right)\right) \text{ by substituting into (3.92)} \\ &= \frac{2T(1)-\sqrt{2}}{n+1} + \frac{1}{n+1}\left(\Theta\left(n^{\frac{5}{2}}\right)\right) \\ &= O(1) + \Theta\left(n^{\frac{3}{2}}\right) \end{aligned}$$

So, $T(n) = \Theta\left(n^{\frac{3}{2}}\right)$. □

Problem 63. *Solve by unfolding*

$$T(n) = \frac{n}{n+1}T(n-1) + \lg n \tag{3.93}$$

Solution:

$$\begin{aligned} T(n) &= \frac{n}{n+1}T(n-1) + \lg n \\ &= \frac{n}{n+1}\left(\frac{n-1}{n}T(n-2) + \lg(n-1)\right) + \lg n \\ &= \frac{n-1}{n+1}T(n-2) + \frac{n}{n+1}\lg(n-1) + \lg n \\ &= \frac{n-1}{n+1}\left(\frac{n-2}{n-1}T(n-3) + \lg(n-2)\right) + \frac{n}{n+1}\lg(n-1) + \lg n \\ &= \frac{n-2}{n+1}T(n-3) + \frac{n-1}{n+1}\lg(n-2) + \frac{n}{n+1}\lg(n-1) + \lg n \\ &= \dots \\ &= \underbrace{\frac{2}{n+1}T(1)}_{A} + \underbrace{\frac{3}{n+1}\lg 2 + \frac{4}{n+1}\lg 3 + \dots + \frac{n-1}{n+1}\lg(n-2) + \frac{n}{n+1}\lg(n-1) + \lg n}_{B} \end{aligned}$$

Clearly, $A = O(1)$. Consider B.

$$\begin{aligned}
B &= \frac{3}{n+1}\lg 2 + \frac{4}{n+1}\lg 3 + \ldots + \frac{n-1}{n+1}\lg(n-2) + \frac{n}{n+1}\lg(n-1) + \frac{n+1}{n+1}\lg n \\
&= \frac{1}{n+1}\left(3\lg 2 + 4\lg 3 + \ldots + (n-1)\lg(n-2) + n\lg(n-1) + (n+1)\lg n\right) \\
&= \frac{1}{n+1}\left(\lg 2^3 + \lg 3^4 + \ldots + \lg(n-2)^{(n-1)} + \lg(n-1)^n + \lg n^{(n+1)}\right) \\
&= \frac{1}{n+1}\lg\left(2^3.3^4\ldots(n-2)^{(n-1)}.(n-1)^n.n^{(n+1)}\right) \\
&= \frac{1}{n+1}\lg\left(\frac{2^{n+1}}{2^{n-2}}\frac{3^{n+1}}{3^{n-3}}\ \cdots\ \frac{(n-2)^{n+1}}{(n-2)^2}\frac{(n-1)^{n+1}}{n-1}\frac{n^{n+1}}{n^0}\right) \\
&= \frac{1}{n+1}\lg\left(\frac{2^{n+1}3^{n+1}\ldots(n-2)^{n+1}(n-1)^{n+1}n^{n+1}}{2^{n-2}3^{n-3}\ldots(n-2)^2(n-1)^1n^0}\right)
\end{aligned}$$

Define that

$$f(n) = 2^{n+1}3^{n+1}\ldots(n-2)^{n+1}(n-1)^{n+1}n^{n+1} \tag{3.94}$$

$$g(n) = 2^{n-2}3^{n-3}\ldots(n-2)^2(n-1)^1n^0 \tag{3.95}$$

so

$$B = \frac{1}{n+1}\lg\left(\frac{f(n)}{g(n)}\right) = \frac{\lg f(n)}{n+1} - \frac{\lg g(n)}{n+1}$$

Using the notations (1.11) and (1.12) on page 2, we claim that $f(n) \succ g(n)$. To see why this is true, note that both $f(n)$ and $g(n)$ have $n-1$ factors and compare the factors in the order in which they appear in (3.94) and (3.95).

$$\begin{array}{lccccccc}
f(n) = & 2^{n+1} & 3^{n+1} & \cdots & (n-2)^{n+1} & (n-1)^{n+1} & n^{n+1} \\
 & \shortmid\!\Upsilon & \shortmid\!\Upsilon & & \shortmid\!\Upsilon & \shortmid\!\Upsilon & \Upsilon \\
g(n) = & 2^{n-2} & 3^{n-3} & \cdots & (n-2)^2 & (n-1)^1 & n^0
\end{array}$$

Now it is obvious that $f(n) \succ g(n)$. Then by (1.37) on page 9, $\lg f(n) \succeq \lg g(n)$, therefore

$$\frac{\lg f(n)}{n+1} \succeq \frac{\lg g(n)}{n+1}$$

It follows that

$$B = \Theta\left(\frac{\lg f(n)}{n+1}\right)$$

It is clear from (3.94) that $f(n) = (n!)^{n+1}$. Therefore,

$$\lg f(n) = \lg(n!)^{n+1} = (n+1)\lg n! = (n+1)\Theta(n\lg n)$$

It follows that

$$B = \Theta(n\lg n)$$

Recall that $T(n) = A + B$ and $A = O(1)$. We conclude that

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

□

Problem 64. *Solve*

$$T(1) = \Theta(1) \tag{3.96}$$
$$T(2) = \Theta(1) \tag{3.97}$$
$$T(n) = T(n-1).T(n-2) \tag{3.98}$$

Solution:
Unlike the problems we encountered so far, the aymptotic growth rate of $T(n)$ in this problem depends on the concrete values of the constants in (3.96) and (3.97). It is easy to see that if $T(1) = T(2) = 1$ then $T(n) = 1$ for all positive n. So let us postulate that

$$T(1) = c \tag{3.99}$$
$$T(2) = d \tag{3.100}$$

where c and d are some positive constants. Then

$$\begin{aligned}
T(3) &= T(2).T(1) = cd \\
T(4) &= T(3).T(2) = cd^2 \\
T(5) &= T(4).T(3) = c^2d^3 \\
T(6) &= T(5).T(4) = c^3d^5 \\
T(7) &= T(6).T(5) = c^5d^8 \\
&\dots
\end{aligned}$$

The degrees that appear in this sequence look like the Fibonacci number (see the definition on page 91). Indeed, it is trivial to prove by induction that

$$\begin{aligned}
T(1) &= c \\
T(n) &= d^{F_{n-1}} c^{F_{n-2}}, \text{ for all } n > 1
\end{aligned} \tag{3.101}$$

Define

$$\begin{aligned}
a &= c^{\frac{1}{\sqrt{5}}} \\
b &= d^{\frac{1}{\sqrt{5}}}
\end{aligned}$$

and derive

$$\begin{aligned}
T(n) &= \Theta\left(b^{\phi^{n-1}}\right)\Theta\left(a^{\phi^{n-2}}\right) && \text{applying (4.15) on page 91 on (3.101)} \\
&= \Theta\left(b^{\phi^{n-1}} a^{\phi^{n-2}}\right) \\
&= \Theta\left(b^{\phi.\phi^{n-2}} a^{\phi^{n-2}}\right) \\
&= \Theta\left(k^{\phi^{n-2}} a^{\phi^{n-2}}\right) && \text{defining that } b^{\phi} = k \\
&= \Theta\left((ak)^{\phi^{n-2}}\right)
\end{aligned} \tag{3.102}$$

Depending on how detalied analysis we need, we may stop right here. However, we can go on a little further because depending on what a and k are, (3.101) can have dramatically different asymptotic growth.

- If $ak > 1$, $T(n) \xrightarrow[n \to +\infty]{} \infty$.
- If $ak = 1$, $T(n) = 1$ for all positive n, thus $T(n) = \Theta(1)$.
- If $ak < 1$, $T(n) \xrightarrow[n \to +\infty]{} 0$, thus $T(n) = O(1)$.

□

3.2.2 The Master Theorem

There are several theoretical results solving a broad range of recurrences corresponding to divide-and-conquer algorithms that are called master theorems. The one stated below is due to [CLR00]. There is a considerately more powerful master theorem due to Akra and Bazzi [AB98]. See [Lei96] for a detailed explanation.

Theorem 1 (Master Theorem, [CLR00], pp. 62)**.** *Let* $a \geq 1$ *and* $b > 1$ *be constants, let* $k = \lg_b a$*, and let* $f(n)$ *be a positive function. Let*

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$
$$T(1) = \Theta(1)$$

where $\frac{n}{b}$ *is interpreted either as* $\left\lfloor \frac{n}{b} \right\rfloor$ *or* $\left\lceil \frac{n}{b} \right\rceil$*. Then* $T(n)$ *can be bounded asymptotically as follows.*

Case 1 *If* $f(n) = O\left(n^{k-\epsilon}\right)$ *for some positive constant* ϵ *then* $T(n) = \Theta(n^k)$*.*

Case 2 *If* $f(n) = \Theta(n^k)$ *then* $T(n) = \Theta\left(n^k . \lg n\right)$*.*

Case 3 *If both*

1. $f(n) = \Omega\left(n^{k+\epsilon}\right)$ *for some positive constant* ϵ*, and*
2. *for some positive constant* c *and for all sufficiently large* n*,* $a.f\left(\frac{n}{b}\right) \leq c.f(n)$*,*

then $T(n) = \Theta(f(n))$*.* □

Case 3-2 is known as *the regularity condition.*

Note that the condition $f(n) = O\left(n^{k-\epsilon}\right)$ is stronger than $f(n) = o(n^k)$ and $f(n) = \Omega\left(n^{k+\epsilon}\right)$ is stronger than $f(n) = \omega\left(n^k\right)$:

$$f(n) = O(n^{k-\epsilon}) \Rightarrow f(n) = o(n^k)$$
$$f(n) = o(n^k) \not\Rightarrow f(n) = O(n^{k-\epsilon})$$
$$f(n) = \Omega(n^{k+\epsilon}) \Rightarrow f(n) = \omega(n^k)$$
$$f(n) = \omega(n^k) \not\Rightarrow f(n) = \Omega(n^{k+\epsilon})$$

For example, consider that

$$n \lg n = \omega(n) \tag{3.103}$$

$$n \lg n \neq \Omega(n^{1+\epsilon}) \text{ for any } \epsilon > 0 \text{ because } \lg n \neq \Omega(n^{\epsilon}) \text{ by (1.44)} \tag{3.104}$$

$$\frac{n}{\lg n} = o(n) \tag{3.105}$$

$$\frac{n}{\lg n} \neq O(n^{1-\epsilon}) \text{ for any } \epsilon > 0 \text{ because } \frac{1}{\lg n} \neq O(n^{-\epsilon}) \tag{3.106}$$

To see why $\frac{1}{\lg n} \neq O(n^{-\epsilon})$ in (3.106) consider that

$$\lim_{n\to\infty} \frac{\lg n}{n^{\epsilon}} = 0 \Rightarrow \lim_{n\to\infty} \left(\frac{\frac{1}{n^{\epsilon}}}{\frac{1}{\lg n}} \right) = 0 \Rightarrow \frac{1}{n^{\epsilon}} = o\left(\frac{1}{\lg n}\right) \text{ by (1.6)} \Rightarrow$$

$$\frac{1}{\lg n} = \omega\left(\frac{1}{n^{\epsilon}}\right) \text{ by the transpose symmetry}$$

Problem 65. *Solve by the Master Theorem*

$$T(n) = 4T\left(\frac{n}{2}\right) + n$$

Solution:
Using the terminology of the Master Theorem, a is 4, b is 2, thus $\log_b a$ is $\log_2 4 = 2$ and $n^{\log_b a}$ is n^2. The function $f(n)$ is n. The theorem asks us to compare $f(n)$ and $n^{\log_b a}$, which, in the current case, is to compare n with n^2. Clearly, $n = O(n^{2-\epsilon})$ for some $\epsilon > 0$, so Case 1 of the Master Theorem is applicable and $T(n) = n^2$. □

Problem 66. *Solve by the Master Theorem*

$$T(n) = T\left(\frac{2n}{3}\right) + 1$$

Solution:
Rewrite the recurrence as

$$T(n) = 1.T\left(\frac{n}{\frac{3}{2}}\right) + 1$$

Using the terminology of the Master Theorem, a is 1, b is $\frac{3}{2}$, thus $\log_b a$ is $\log_{\frac{3}{2}} 1 = 0$ and $n^{\log_b a}$ is $n^0 = 1$. The function $f(n)$ is n. Clearly, $1 = \Theta(n^0)$, so Case 2 of the Master Theorem is applicable. Assording to it, $T(n) = \Theta(1. \lg n) = \Theta(\lg n)$. □

Problem 67. *Solve*

$$T(n) = 3T\left(\frac{n}{4}\right) + n \lg n$$

Solution:
Using the terminology of the Master Theorem, a is 3, b is 4, thus $\log_b a$ is $\log_4 3$, which is approximately 0.79, and the function $f(n)$ is $n \lg n$. It certainly is true that $n \lg n = \Omega(n^{\log_4 3+\epsilon})$ for some $\epsilon > 0$, for instance $\epsilon = 0.1$. However, we have to check the regularity condition to see if Case 3 of the Master Theorem is aplicable. The regularity condition in this case is:

$$\exists c \text{ such that } 0 < c < 1 \text{ and } 3\frac{n}{4} \lg \frac{n}{4} \leq cn \lg n \text{ for all sufficiently large } n$$

The latter clearly holds for, say, $c = \frac{3}{4}$, therefore Case 3 is applicable and according to it, $T(n) = \Theta(n \lg n)$. □

Problem 68. *Solve*

$$T(n) = 2T\left(\frac{n}{2}\right) + n \lg n$$

Solution:
Let us the try to solve it using the Master Theorem. Using the terminology of the Master Theorem, a is 2 and b is 2, thus $\log_b a$ is $\log_2 2 = 1$, therefore $n^{\log_b a}$ is $n^1 = n$. The function $f(n)$ is $n \lg n$. Let us see if we can classify that problem in one of the three cases of the Master Theorem.

try Case 1 Is it true that $n \lg n = O(n^{1-\epsilon})$ for some $\epsilon > 0$? No, because $n \lg n = \omega(n^1)$.

try Case 2 Is it true that $n \lg n = \Theta(n^1)$? No, because $n \lg n = \omega(n^1)$.

try Case 3 Is it true that $n \lg n = \Omega(n^{1+\epsilon})$ for some $\epsilon > 0$? No, see (3.104).

Therefore this problem cannot be solved using the Master Theorem as stated above. We solve it by Theorem 2 on page 77 and the answer is $T(n) = \Theta(n \lg^2 n)$. □

Problem 69. *Solve*

$$T(n) = 4T\left(\frac{n}{2}\right) + n \tag{3.107}$$

$$T(n) = 4T\left(\frac{n}{2}\right) + n^2 \tag{3.108}$$

$$T(n) = 4T\left(\frac{n}{2}\right) + n^3 \tag{3.109}$$

$$\tag{3.110}$$

Solution:
Using the terminology of the Master Theorem, a is 4 and b is 2, thus $\log_b a$ is $\log_4 2 = 2$, therefore $n^{\log_b a}$ is n^2. With respect to (3.107), it is the case that $n = O(n^{2-\epsilon})$ for some $\epsilon > 0$, therefore the solution of (3.107) is $T(n) = \Theta(n^2)$ by Case 1 of the Master Theorem. With respect to (3.108), it is the case that $n^2 = \Theta(n^2)$, therefore the solution of (3.108) is $T(n) = \Theta(n^2 \lg n)$ by Case 2 of the Master Theorem. With respect to (3.109), it is the case that $n^3 = \Omega(n^{2+\epsilon})$ for some $\epsilon > 0$, therefore the solution of (3.109) is $T(n) = \Theta(n^3)$

by Case 3 of the Master Theorem, *provided* the regularity condition holds. The regularity condition here is

$$\exists c \text{ such that } 0 < c < 1 \text{ and } 4\left(\frac{n}{2}\right)^3 \leq cn^3 \text{ for all sufficiently large } n$$

Clearly that holds for any c such that $\frac{1}{2} \leq c < 1$. Therefore, by Case 3 of the Master Theorem, the solution of (3.109) is $T(n) = \Theta(n^3)$. □

Problem 70. *Solve*

$$T(n) = T\left(\frac{n}{2}\right) + \lg n \tag{3.111}$$

Solution:
Let us try to solve it using the Master Theorem. Using the terminology of the Master Theorem, a is 1 and b is 2, thus $\log_b a$ is $\log_2 1 = 0$, therefore $n^{\log_b a}$ is $n^0 = 1$. The function $f(n)$ is $\lg n$. Let us see if we can classify that problem in one of the three cases of the Master Theorem.

try Case 1 Is it true that $\lg n = O(n^{0-\epsilon})$ for some $\epsilon > 0$? No, because $\lg n$ is an increasing function and $n^{-\epsilon} = \frac{1}{n^\epsilon}$ is a decreasing one.

try Case 2 Is it true that $\lg n = \Theta(n^0)$? No, because $\lg n = \omega(n^0)$.

try Case 3 Is it true that $\lg n = \Omega(n^{0+\epsilon})$ for some $\epsilon > 0$? No, see (1.44) on page 11.

So the Master Theorem is not applicable and we seek other methods for solving. Substitute n by 2^m, *i.e.* $m = \lg n$ and $m = \lg n$. Then (3.111) becomes

$$T(2^m) = T\left(2^{m-1}\right) + m \tag{3.112}$$

Further substitute $T(2^m)$ by $S(m)$ and (3.112) becomes

$$S(m) = S(m-1) + m \tag{3.113}$$

But that recurrence is the same as (3.18), therefore its solution is $S(m) = \Theta(m^2)$. Let us go back now to the original n and $T(n)$.

$$S(m) = \Theta(m^2) \;\Leftrightarrow\; T(2^m) = \Theta(\lg^2 n) \;\Leftrightarrow\; T(n) = \Theta(\lg^2 n)$$

□

Problem 71. *Solve by the Master Theorem*

$$T(n) = 2T\left(\frac{n}{2}\right) + n^3 \tag{3.114}$$

$$T(n) = T\left(\frac{9n}{10}\right) + n \tag{3.115}$$

$$T(n) = 16T\left(\frac{n}{4}\right) + n^2 \tag{3.116}$$

$$T(n) = 7T\left(\frac{n}{3}\right) + n^2 \tag{3.117}$$

$$T(n) = 7T\left(\frac{n}{2}\right) + n^2 \tag{3.118}$$

$$T(n) = 2T\left(\frac{n}{4}\right) + \sqrt{n} \tag{3.119}$$

$$T(n) = 4T\left(\frac{n}{2}\right) + n^2\sqrt{n} \tag{3.120}$$

$$T(n) = 8T\left(\frac{n}{2}\right) + n^3 \tag{3.121}$$

$$T(n) = 3T\left(\frac{n}{2}\right) + 2n^2 \tag{3.122}$$

$$T(n) = 3T\left(\frac{n}{2}\right) + n\lg n \tag{3.123}$$

Solution:
(3.114): as $n^3 = \Omega\left(n^{\log_2 2+\epsilon}\right)$ for some $\epsilon > 0$, we classify the problem into Case 3 of the Master Theorem. To apply Case 3, we have to check the regularity condition holds. Namely, there is a constant c such that $0 < c < 1$ and $2\left(\frac{n}{2}\right)^3 \leq cn^3 \Leftrightarrow \frac{1}{4} \leq c$. So, any c such that $\frac{1}{4} \leq c < 1$ will do, therefore the regularity condition holds, therefore Case 3 is applicable, therefore $T(n) = \Theta(n^3)$.

(3.115): rewrite the recurrence as $T(n) = 1.T\left(\frac{n}{\frac{10}{9}}\right) + n$. As $n = \Omega\left(n^{\left(\log_{\frac{10}{9}} 1\right)+\epsilon}\right)$ for some $\epsilon > 0$, we classify the problem into Case 3 of the Master Theorem. To apply Case 3, we have to check the regularity condition holds. Namely, there is a constant c such that $0 < c < 1$ and $1\left(\frac{n}{\frac{10}{9}}\right) \leq cn \Leftrightarrow \frac{9}{10} \leq c$. So, any c such that $\frac{9}{10} \leq c < 1$ will do, therefore the regularity condition holds, therefore Case 3 is applicable, therefore $T(n) = \Theta(n)$.

(3.116): As $n^2 = \Theta\left(n^{\log_4 16}\right)$, we classify the problem into Case 2 of the Master Theorem and so $T(n) = n^2 \lg n$.

(3.117): as $n^2 = \Omega\left(n^{\log_3 7+\epsilon}\right)$ for some $\epsilon > 0$, we classify the problem into Case 3 of the Master Theorem. To apply Case 3, we have to check the regularity condition holds. Namely, there is a constant c such that $0 < c < 1$ and $7\left(\frac{n}{3}\right)^2 \leq cn^2 \Leftrightarrow \frac{7}{9} \leq c$. So, any c such that $\frac{7}{9} \leq c < 1$ will do, therefore the regularity condition holds, therefore Case 3 is applicable, therefore $T(n) = \Theta(n^2)$.

(3.118): as $n^2 = O\left(n^{\log_2 7-\epsilon}\right)$ for some $\epsilon > 0$, we classify the problem into Case 1 of the Master Theorem and so $T(n) = \Theta\left(n^{\log_2 7}\right)$.

(3.119): as $\sqrt{n} = \Theta\left(n^{\log_4 2}\right)$, we classify the problem into Case 2 of the Master Theorem and so $T(n) = \Theta(\sqrt{n} \lg n)$.

(3.120): as $n^{\frac{5}{2}} = \Omega\left(n^{\log_2 4+\epsilon}\right)$ for some $\epsilon > 0$, we classify the problem into Case 3 of the Master Theorem. To apply Case 3, we have to check the regularity condition holds. Namely, there is a constant c such that $0 < c < 1$ and $4\left(\frac{n}{2}\right)^{\frac{5}{2}} \leq cn^{\frac{5}{2}} \Leftrightarrow \frac{1}{\sqrt{2}} \leq c$. So, any c such that $\frac{1}{\sqrt{2}} \leq c < 1$ will do, therefore the regularity condition holds, therefore Case 3 is applicable, therefore $T(n) = \Theta(n^2\sqrt{n})$.

(3.121): As $n^3 = \Theta\left(n^{\log_8 2}\right)$, we classify the problem into Case 2 of the Master Theorem and so $T(n) = n^3 \lg n$.

(3.122): as $2n^2 = \Omega\left(n^{\log_2 3+\epsilon}\right)$ for some $\epsilon > 0$, we classify the problem into Case 3 of the Master Theorem. To apply Case 3, we have to check the regularity condition holds. Namely, there is a constant c such that $0 < c < 1$ and $3\left(2\left(\frac{n}{2}\right)^2\right) \leq c2n^2 \Leftrightarrow 3 \leq 4c$. So, any c such that $\frac{3}{4} \leq c < 1$ will do, therefore the regularity condition holds, therefore Case 3 is applicable, therefore $T(n) = \Theta(2n^2) = \Theta(n^2)$.

(3.123): as $n \lg n = O\left(n^{\log_2 3-\epsilon}\right)$ for some $\epsilon > 0$, we classify the problem into Case 1 of the Master Theorem and so $T(n) = \Theta\left(n^{\log_2 3}\right)$. □

The following result extends Case 2 of the Master Theorem.

Theorem 2. *Under the premises of Theorem 1, assume*

$$f(n) = \Theta(n^k \lg^t n) \tag{3.124}$$

for some constant $t \geq 0$. *Then*

$$T(n) = \Theta(n^k \lg^{t+1} n)$$

Proof:
Theorem 1 itself is not applicable because the recurrence for the said $f(n)$ cannot be classified into any of the three cases there. To solve the problem we use unfolding. For simplicity we assume that n is an exact power of b, *i.e.* $n = b^m$ for some integer $m > 0$. The same technique is used in [CLR00] for proving the Master Theorem: first prove it for exact powers of b and then prove the result holds for any positive n. Here we limit our proof to the case that n is an exact power of b and leave it to the reader to generalise for any positive n.

Assume that the logarithm in (3.124) is base-b and note we can rewrite what is inside the Θ-notation on the right-hand side of (3.124) in the following way:

$$n^k \log_b^t n = n^{\log_b a} (\log_b b^m)^t = b^{(m \log_b a)} m^t = b^{(\log_b a^m)} m^t = a^m m^t \tag{3.125}$$

Then (3.124) is equivalent to saying that

$$c_1 a^m m^t \leq f(b^m) \leq c_2 a^m m^t$$

for some positive constants c_1 and c_2 and all sufficiently large values of m. However, for the sake of simplicity, we will assume in the remainder of the proof that

$$f(b^m) = a^m m^t \tag{3.126}$$

The reader is invited to construct a proof for the general case.

By the definition of the Master Theorem, $T(n) = aT\left(\frac{n}{b}\right) + f(n)$. Using (3.126) we rewrite that as follows.

$$\begin{aligned}
T(b^m) &= aT\left(\frac{b^m}{b}\right) + a^m m^t \\
&= aT(b^{m-1}) + a^m m^t \;\Leftrightarrow \\
S(m) &= aS(m-1) + a^m m^t \qquad \text{substituting } T(b^m) \text{ with } S(m) \\
&= a\big(aS(m-2) + a^{m-1}(m-1)^t\big) + a^m m^t \\
&= a^2 S(m-2) + a^m (m-1)^t + a^m m^t \\
&= a^2\big(aS(m-3) + a^{m-2}(m-2)^t\big) + a^m (m-1)^t + a^m m^t \\
&= a^3 S(m-3) + a^m (m-2)^t + a^m (m-1)^t + a^m m^t \\
&\dots \\
&= a^{m-1} S(1) + a^m 2^t + a^m 3^t + \ldots + a^m (m-2)^t + a^m (m-1)^t + a^m m^t \\
&= a^{m-1} S(1) - a^m + a^m \underbrace{(1^t + 2^t + 3^t + \ldots + (m-2)^t + (m-1)^t + m^t)}_{\Theta(m^{t+1}) \text{ by (4.20) on page 92}} \\
&= a^{m-1} S(1) - a^m + a^m \Theta(m^{t+1}) \\
&= a^{m-1} S(1) - a^m + \Theta(a^m m^{t+1}) \qquad (3.127)
\end{aligned}$$

But (3.127) is $\Theta(a^m m^{t+1})$ because $a^m m^{t+1} = \omega(|a^{m-1}S(1) - a^m|)$. So,

$$S(m) = \Theta(a^m m^{t+1}) \;\Leftrightarrow\; T(n) = \Theta\left(a^{\log_b n} (\log_b n)^{t+1}\right)$$

Having in mind that $a^{\log_b n} = n^{\log_b a}$ and $\log_b n = \Theta(\lg n)$, we conclude that

$$T(n) = \Theta\left(n^{\log_b a} \lg^{t+1} n\right)$$

□

Problem 72. *Solve*

$$T(n) = 2T\left(\frac{n}{2}\right) + \frac{n}{\lg n}$$

Solution:
Let us the try to solve it using the Master Theorem. Using the terminology of the Master Theorem, a is 2 and b is 2, thus $\log_b a$ is $\log_2 2 = 1$, therefore $n^{\log_b a}$ is $n^1 = n$. The function $f(n)$ is $\frac{n}{\lg n}$. Let us see if we can classify that problem in one of the three cases of the Master Theorem.

try Case 1 Is it true that $\frac{n}{\lg n} = O(n^{1-\epsilon})$ for some $\epsilon > 0$? No, see (3.106) on page 73.

try Case 2 Is it true that $\frac{n}{\lg n} = \Theta(n^1)$? No, because $n \lg n = o(n^1)$.

try Case 3 Is it true that $\frac{n}{\lg n} = \Omega(n^{1+\epsilon})$ for some $\epsilon > 0$? No, because $n \lg n = o(n^1)$.

Therefore this problem cannot be solved using the Master Theorem as stated above. Furthermore, Theorem 2 on page 77 cannot be applied either because it is not true that $\frac{n}{\lg n} = \Theta(n^{\log_2 2} \lg^t(n))$ for any $t \geq 0$.

We solve the problem by unfolding.

$$\begin{aligned}
T(n) &= 2T\left(\frac{n}{2}\right) + \frac{n}{\lg n} \\
&= 2\left(2T\left(\frac{n}{4}\right) + \frac{\frac{n}{2}}{\lg \frac{n}{2}}\right) + \frac{n}{\lg n} \\
&= 4T\left(\frac{n}{4}\right) + \frac{n}{(\lg n) - 1} + \frac{n}{\lg n} \\
&= 4\left(2T\left(\frac{n}{8}\right) + \frac{\frac{n}{4}}{\lg \frac{n}{4}}\right) + \frac{n}{(\lg n) - 1} + \frac{n}{\lg n} \\
&= 8T\left(\frac{n}{8}\right) + \frac{n}{(\lg n) - 2} + \frac{n}{(\lg n) - 1} + \frac{n}{\lg n} \\
&\ldots \\
&= nT(1) + \frac{n}{2} + \frac{n}{3} + \ldots + \frac{n}{(\lg n) - 2} + \frac{n}{(\lg n) - 1} + \frac{n}{\lg n} \\
&= \underbrace{nT(1) - n}_{A} + \underbrace{n\left(\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{(\lg n) - 2} + \frac{1}{(\lg n) - 1} + \frac{1}{\lg n}\right)}_{B}
\end{aligned}$$

Clearly, $|A| = O(n)$. Now observe that $B = n.H_{\lg n}$ because inside the parentheses is the $(\lg n)^{\text{th}}$ partial sum of the harmonic series (see 4.16 on page 91). By (4.17), $H_{\lg n} = \Theta(\lg \lg n)$, therefore $B = \Theta(n \lg \lg n)$, therefore $T(n) = \Theta(n \lg \lg n)$. □

Chapter 4

Appendix

Problem 73. *Find a closed formula for*

$$\sum_{k=0}^{n} 2^k k$$

Solution:
Let

$$S_n = \sum_{k=0}^{n} 2^k k$$

Then

$$S_n + (n+1)2^{n+1} = \sum_{k=0}^{n} 2^k k + (n+1)2^{n+1} = \sum_{k=0}^{n} 2^{k+1}(k+1) = 2\sum_{k=0}^{n} 2^k k + 2\sum_{k=0}^{n} 2^k$$

Since $\sum_{k=0}^{n} 2^k = 2^{n+1} - 1$,

$$S_n + (n+1)2^{n+1} = \underbrace{2\sum_{k=0}^{n} 2^k k}_{2S_n} + 2(2^{n+1} - 1) = 2S_n + 2.2^{n+1} - 2$$

Then

$$S_n = n2^{n+1} + 2^{n+1} - 2.2^{n+1} + 2 = n2^{n+1} - 2^{n+1} + 2$$

So,

$$S_n = (n-1)2^{n+1} + 2 \tag{4.1}$$

□

Problem 74. *Find a closed formula for*

$$\sum_{k=0}^{n} 2^k k^2$$

Solution:
Let

$$S_n = \sum_{k=0}^{n} 2^k k^2$$

Then

$$\begin{aligned} S_n + 2^{n+1}(n+1)^2 &= \sum_{k=0}^{n} 2^k k^2 + 2^{n+1}(n+1)^2 = \sum_{k=0}^{n} 2^{k+1}(k+1)^2 \\ &= 2\sum_{k=0}^{n} 2^k(k^2+2k+1) \\ &= \underbrace{2\sum_{k=0}^{n} 2^k k^2}_{2S_n} + \underbrace{4\sum_{k=0}^{n} 2^k k}_{4(n-1)2^{n+1}+8} + \underbrace{2\sum_{k=0}^{n} 2^k}_{2.2^{n+1}-2} \end{aligned}$$

Then

$$S_n + n^2 2^{n+1} + 2n2^{n+1} + 2.2^{n+1} = 2S_n + 4n2^{n+1} - 4.2^{n+1} + 8 + 2.2^{n+1} - 2$$

So,

$$S_n = n^2 2^{n+1} - 2n2^{n+1} + 4.2^{n+1} - 6 \tag{4.2}$$

□

Problem 75. *Find a closed formula for the sum of the first* n *odd numbers*

$$S_n = 1 + 3 + 5 + \ldots + 2n - 1$$

Solution:
It is trivial to prove by induction on n that $S_n = n^2$.
Basis: $S_1 = 1^2$.
Induction hypothesis: assume $S_n = n^2$.
Induction step:

$$\begin{aligned} S_{n+1} &= 1 + 3 + 5 + \ldots + 2n - 1 + 2n + 1 \\ &= S_n + 2n + 1 \qquad \text{by definition} \\ &= n^2 + 2n + 1 \qquad \text{by the induction hypothesis} \\ &= (n+1)^2 \end{aligned}$$

Indeed,

$$S_n = n^2 \tag{4.3}$$

There is a geometric proof of the same fact, illustrated on Figure 4.1. □

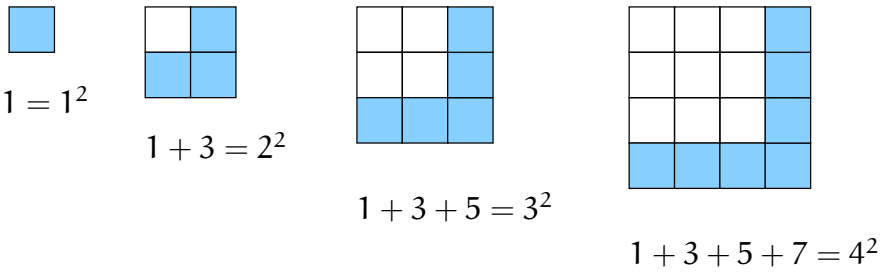

Figure 4.1: A geometric proof that the sum of the first n odd numbers is the n^{th} square n^2.

Problem 76. *Find a closed formula for*

$$\sum_{i=1}^{n} \left\lfloor \sqrt{i} \right\rfloor$$

Solution:
To gain some intuition, let us write down the sum explicitly, *i.e.* all the terms, for some small n, say $n = 17$. For clarity put boxes around the terms whose positions are perfect squares, *i.e.* around the first, fourth, ninth, and sixtieth term.

$$\sum_{i=1}^{17} \left\lfloor \sqrt{i} \right\rfloor = \underbrace{\boxed{1} + 1 + 1}_{\text{run } 1} + \underbrace{\boxed{2} + 2 + 2 + 2 + 2}_{\text{run } 2} + \underbrace{\boxed{3} + 3 + 3 + 3 + 3 + 3 + 3}_{\text{run } 3} + \underbrace{\boxed{4} + 4}_{\text{run } 4}$$

The pattern is clear: the sum is the first n, in this case $n = 17$, terms of a series whose terms are the consecutive positive integers grouped in *runs*, run j being the sum of $2j + 1$ in number j's. Naturally, each run starts at a term whose position in the series is a perfect square: run 1 starts at position 1, run 2 starts at position 4, run 3 starts at position 9, *etc.* Problem 75 explains why the runs, except possibly for the last run, have lengths that are the consecutive odd numbers—since the first j odd numbers sum precisely to a perfect square, *viz.* j^2, it follows the difference between the two consecutive perfect squares $(j + 1)^2 - j^2$ is an odd number, *viz.* $2j + 1$.

The run with the largest number can be incomplete, as is the case when $n = 17$—run number 4 has only two terms. Let us call the number of complete runs, *i.e.* the ones that have all the terms, k_n. For instance, $k_{17} = 3$. We claim that

$$k_n = \lfloor \sqrt{n + 1} \rfloor - 1$$

To see why, imagine that n decreases one by one and think of the moment when k_n decreases. That is not when n becomes a perfect square minus one but when n becomes a perfect square minus two. For instance, $k_{15} = 3$ but $k_{14} = 2$. Hence we have $\sqrt{n + 1}$, not $\sqrt{n}$.

Having all that in mind we break the desired sum down into two sums:

$$\sum_{i=1}^{n} \left\lfloor \sqrt{i} \right\rfloor = S_1 + S_2$$

where S_1 is the sum of the terms of the complete runs and S_2, of the incomplete run. $S_2 = 0$ if and only if n is a perfect square minus one. More precisely, if we denote the number of

terms in S_2 by l_n,

$$l_n = n - \lfloor\sqrt{n+1}\rfloor^2 + 1$$

For instance, $l_{17} = 2$ as seen above and indeed $17 - \lfloor\sqrt{17+1}\rfloor^2 + 1 = 17 - 4^2 + 1 = 2$; $l_{15} = 0$ as seen above and indeed $15 - \lfloor\sqrt{15+1}\rfloor^2 + 1 = 15 - 4^2 + 1 = 0$.

Let us first compute S_1.

$$\begin{aligned}
S_1 &= 1.3 + 2.5 + 3.7 + 4.9 + 5.11 + \ldots + k(n)(2k(n)+1) \\
&= \sum_{i=1}^{k(n)} i(2i+1) \\
&= 2\sum_{i=1}^{k(n)} i^2 + \sum_{i=1}^{k(n)} i \\
&= 2\frac{k(n).(k(n)+1).(2k(n)+1)}{6} + \frac{k(n).(k(n)+1)}{2} \quad \text{by (4.21) and (4.22)} \\
&= k(n).(k(n)+1)\left(\frac{4k(n)+2}{6} + \frac{3}{6}\right) \\
&= \frac{1}{6}k(n).(k(n)+1).(4k(n)+5) \\
&= \frac{1}{6}(\lfloor\sqrt{n+1}\rfloor - 1)(\lfloor\sqrt{n+1}\rfloor - 1 + 1)(4\lfloor\sqrt{n+1}\rfloor - 4 + 5) \\
&= \frac{1}{6}(\lfloor\sqrt{n+1}\rfloor - 1)\lfloor\sqrt{n+1}\rfloor(4\lfloor\sqrt{n+1}\rfloor + 1)
\end{aligned}$$

Clearly, $S_1 = \Theta\left(n^{\frac{3}{2}}\right)$. S_2 is easier to compute, it has $l(n)$ terms, each term being $k(n)+1$.

$$\begin{aligned}
S_2 &= l_n(k_n + 1) \\
&= (n - \lfloor\sqrt{n+1}\rfloor^2 + 1)(\lfloor\sqrt{n+1}\rfloor - 1 + 1) \\
&= (n - \lfloor\sqrt{n+1}\rfloor^2 + 1)\lfloor\sqrt{n+1}\rfloor
\end{aligned}$$

Clearly, $S_2 = O\left(n^{\frac{3}{2}}\right)$, therefore $S_1 + S_2 = \Theta\left(n^{\frac{3}{2}}\right) + O\left(n^{\frac{3}{2}}\right) = \Theta\left(n^{\frac{3}{2}}\right)$.

Let us denote $\lfloor\sqrt{n+1}\rfloor$ by $\widetilde{n}$. It follows that

$$\begin{aligned}
S_1 &= \frac{\widetilde{n}(\widetilde{n}-1)(4\widetilde{n}+1)}{6} \\
S_2 &= (n - \widetilde{n}^2 + 1)\widetilde{n}
\end{aligned}$$

$$\sum_{i=1}^{n} \left\lfloor\sqrt{i}\right\rfloor = \widetilde{n}\left(\frac{(\widetilde{n}-1)(4\widetilde{n}+1)}{6} + (n - \widetilde{n}^2 + 1)\right) \tag{4.4}$$

and

$$\sum_{i=1}^{n} \left\lfloor\sqrt{i}\right\rfloor = \Theta\left(n^{\frac{3}{2}}\right) \tag{4.5}$$

□

Problem 77. *Find a closed formula for*

$$\sum_{i=1}^{n} \left\lceil \sqrt{i} \right\rceil$$

Solution:
Let us start with a small example as in Problem 76, say for $n = 17$. For clarity put boxes around the terms whose positions are perfect squares, *i.e.* around the first, fourth, ninth, and sixtienth term.

$$\sum_{i=1}^{17} \left\lceil \sqrt{i} \right\rceil = \underbrace{\boxed{1}}_{\text{run 1}} + \underbrace{2+2+\boxed{2}}_{\text{run 2}} + \underbrace{3+3+3+3+\boxed{3}}_{\text{run 3}} + \underbrace{4+4+4+4+4+4+\boxed{4}}_{\text{run 4}} + \underbrace{5}_{\text{run5}}$$

The pattern is quite similar to the one in Problem 76. We sum the first n terms of a series whose terms are the consecutive positive integers grouped in runs, run j being the sum of $2j-1$ in number j's.

The run with the largest number can be incomplete. For instance, if $n = 17$ then run number 5 has only one term. Let us call the number of complete runs, *i.e.* the ones that have all the terms, s_n. For instance, $s_{17} = 4$. It is obvious that

$$s_n = \lfloor \sqrt{n} \rfloor$$

We break the desired sum down into two sums:

$$\sum_{i=1}^{n} \left\lfloor \sqrt{i} \right\rfloor = S_1 + S_2$$

where S_1 is the sum of the terms of the complete runs and S_2, of the incomplete run. $S_2 = 0$ if and only if n is a perfect square. We denote the number of terms in S_2 by t_n.

$$t_n = n - \lfloor \sqrt{n} \rfloor^2$$

For instance, $t_{17} = 1$ as seen above and indeed $17 - \lfloor \sqrt{17} \rfloor^2 = 17 - 4^2 = 1$; $t_{16} = 0$ as seen above and indeed $16 - \lfloor \sqrt{16} \rfloor^2 = 16 - 4^2 = 0$.

Let us compute S_1.

$$\begin{aligned}
S_1 &= 1.1 + 2.3 + 3.5 + 4.7 + 5.9 + \ldots + s_n(2s_n - 1) \\
&= \sum_{i=1}^{s_n} i(2i-1) \\
&= 2\sum_{i=1}^{s_n} i^2 - \sum_{i=1}^{s_n} i \\
&= 2\frac{s_n.(s_n+1).(2s_n+1)}{6} - \frac{s_n.(s_n+1)}{2} \quad \text{by (4.21) and (4.22)} \\
&= s_n.(s_n+1)\left(\frac{4s_n+2}{6} - \frac{3}{6}\right) \\
&= \frac{1}{6}s_n.(s_n+1).(4s_n-1) \\
&= \frac{1}{6}(\lfloor \sqrt{n} \rfloor)(\lfloor \sqrt{n} \rfloor + 1)(4\lfloor \sqrt{n} \rfloor - 1)
\end{aligned}$$

Clearly, $S_1 = \Theta\left(n^{\frac{3}{2}}\right)$. Now we compute S_2. It has t_n terms, each term being $s_n + 1$.

$$\begin{aligned} S_2 &= t_n.(s_n + 1) \\ &= (n - \lfloor\sqrt{n}\rfloor^2)(\lfloor\sqrt{n}\rfloor + 1) \end{aligned}$$

Clearly, $S_2 = O\left(n^{\frac{3}{2}}\right)$, therefore $S_1 + S_2 = \Theta\left(n^{\frac{3}{2}}\right) + O\left(n^{\frac{3}{2}}\right) = \Theta\left(n^{\frac{3}{2}}\right)$.

It follows that

$$\sum_{i=1}^{n} \left\lceil\sqrt{i}\right\rceil = (\lfloor\sqrt{n}\rfloor + 1)\left(\frac{\lfloor\sqrt{n}\rfloor(4\lfloor\sqrt{n}\rfloor - 1)}{6} + n - \lfloor\sqrt{n}\rfloor^2\right) \tag{4.6}$$

and

$$\sum_{i=1}^{n} \left\lceil\sqrt{i}\right\rceil = \Theta\left(n^{\frac{3}{2}}\right) \tag{4.7}$$

□

Problem 78. *Find a closed formula for*

$$\sum_{i=1}^{n} i\left\lfloor\sqrt{i}\right\rfloor$$

Solution:
The line of reasoning is very similar to the one in Problem 76. We sum the first n terms of a series, the series being the one mentioned in the solution of Problem 76 with each term multiplied by its position. Consider for example $n = 17$. The terms whose positions are perfect squares are boxed.

$$\sum_{i=1}^{17} i\left\lfloor\sqrt{i}\right\rfloor = \underbrace{\boxed{1} + 2 + 3}_{\text{run 1}} + \underbrace{\boxed{8} + 10 + 12 + 14 + 16}_{\text{run 2}} + \underbrace{\boxed{27} + 30 + 33 + 36 + 39 + 42 + 45}_{\text{run 3}} + \underbrace{\boxed{64} + 68}_{\text{run 4}}$$

Unlike Problem 76, now the runs consist of those consecutive terms whose differences are equal (and equal to the number of the run). Just as in Problem 76, all the runs but the last one are complete, the last run being either complete or incomplete. We denote the number of the complete runs with k_n and the number of terms in the incomplete run by l_n. It is the case that

$$\begin{aligned} k_n &= \lfloor\sqrt{n+1}\rfloor - 1 \\ l_n &= n - \lfloor\sqrt{n+1}\rfloor^2 + 1 \end{aligned}$$

the reasoning being exactly the same as in Problem 76. We break the desired sum down into two sums:

$$\sum_{i=1}^{n} i\left\lfloor\sqrt{i}\right\rfloor = S_1 + S_2$$

where S_1 is the sum of the terms of the complete runs and S_2, of the incomplete run.

Let us first compute S_1.

$$\begin{aligned}
S_1 &= 1.(1+2+3) + 2.(4+5+6+7+8) + 3.(9+10+11+12+13+14+15) \\
&\quad + 4.(16+17+18+19+20+21+22+23+24) \\
&\quad + 5.(25+26+27+28+29+30+31+32+33+34+35) \\
&\quad + \dots \\
&\quad + k_n\big(k_n^2 + (k_n^2+1) + (k_n^2+2) + \dots + \underbrace{((k_n+1)^2-1)}_{k_n^2+2k_n}\big) \\
&= \sum_{i=1}^{k_n} i \sum_{j=i^2}^{i^2+2i} j \\
&= \sum_{i=1}^{k_n} i \left(\sum_{j=1}^{i^2+2i} j - \sum_{j=1}^{i^2-1} j \right) \\
&= \sum_{i=1}^{k_n} i \left(\frac{(i^2+2i)(i^2+2i+1)}{2} - \frac{(i^2-1)i^2}{2} \right) \\
&= \frac{1}{2} \sum_{i=1}^{k_n} i \left(i^4 + 2i^3 + i^2 + 2i^3 + 4i^2 + 2i - i^4 + i^2 \right) \\
&= \frac{1}{2} \sum_{i=1}^{k_n} i \left(4i^3 + 6i^2 + 2i \right) \\
&= 2\sum_{i=1}^{k_n} i^4 + 3\sum_{i=1}^{k_n} i^3 + \sum_{i=1}^{k_n} i^2 \quad \text{apply (4.22), (4.23), and (4.24)} \\
&= 2\frac{k_n(k_n+1)(2k_n+1)(3k_n^2+3k_n-1)}{30} + 3\frac{k_n^2(k_n+1)^2}{4} + \frac{k_n(k_n+1)(2k_n+1)}{6} \\
&= \frac{k_n(k_n+1)}{2} \left(\frac{(4k_n+2)(3k_n^2+3k_n-1)}{15} + \frac{3k_n(k_n+1)}{2} + \frac{2k_n+1}{3} \right) \\
&= \frac{k_n(k_n+1)}{60} \left((8k_n+4)(3k_n^2+3k_n-1) + 45k_n(k_n+1) + 20k_n + 10 \right) \\
&= \frac{k_n(k_n+1)}{60} \left(24k_n^3 + 24k_n^2 - 8k_n + 12k_n^2 + 12k_n - 4 + 45k_n^2 + 45k_n + 20k_n + 10 \right) \\
&= \frac{k_n(k_n+1)}{60} \left(24k_n^3 + 81k_n^2 + 69k_n + 6 \right) \\
&= \frac{k_n(k_n+1)(8k_n^3+27k_n^2+23k_n+2)}{20} \qquad (4.8)
\end{aligned}$$

Substitute k_n with $\lfloor\sqrt{n+1}\rfloor - 1$ in (4.8) to obtain

$$\begin{aligned}
S_1 &= \frac{1}{20}\lfloor\sqrt{n+1}\rfloor(\lfloor\sqrt{n+1}\rfloor - 1)\big(8(\lfloor\sqrt{n+1}\rfloor - 1)^3 + \\
&\qquad 27(\lfloor\sqrt{n+1}\rfloor - 1)^2 + 23(\lfloor\sqrt{n+1}\rfloor - 1) + 2\big) \\
&= \frac{1}{20}\lfloor\sqrt{n+1}\rfloor(\lfloor\sqrt{n+1}\rfloor - 1)\big(8\lfloor\sqrt{n+1}\rfloor^3 - 24\lfloor\sqrt{n+1}\rfloor^2 + 24\lfloor\sqrt{n+1}\rfloor - 8 \\
&\qquad 27\lfloor\sqrt{n+1}\rfloor^2 - 54\lfloor\sqrt{n+1}\rfloor + 27 + 23\lfloor\sqrt{n+1}\rfloor - 23 + 2\big) \\
&= \frac{1}{20}\lfloor\sqrt{n+1}\rfloor(\lfloor\sqrt{n+1}\rfloor - 1)\big(8\lfloor\sqrt{n+1}\rfloor^3 + 3\lfloor\sqrt{n+1}\rfloor^2 - 7\lfloor\sqrt{n+1}\rfloor - 2\big)
\end{aligned}$$

Clearly, $S_1 = \Theta\left(n^{\frac{5}{2}}\right)$. Now we compute S_2. It has l_n terms, the first term is $(k_n+1)^3$, and the difference between every two consecutive terms is (k_n+1).

$$\begin{aligned}
S_2 &= \sum_{i=1}^{l_n}(k_n+1)^3 + (i-1)(k_n+1) \\
&= (k_n+1)^3\sum_{i=1}^{l_n} 1 + (k_n+1)\sum_{i=1}^{l_n}(i-1) \\
&= (k_n+1)^3 l_n + \frac{(k_n+1)(l_n-1)l_n}{2} \\
&= \lfloor\sqrt{n+1}\rfloor^3(n - \lfloor\sqrt{n+1}\rfloor^2 + 1) + \frac{\lfloor\sqrt{n+1}\rfloor(n - \lfloor\sqrt{n+1}\rfloor^2)(n - \lfloor\sqrt{n+1}\rfloor^2 + 1)}{2}
\end{aligned}$$

Clearly, $S_2 = O\left(n^{\frac{5}{2}}\right)$, therefore $S_1 + S_2 = \Theta\left(n^{\frac{5}{2}}\right) + O\left(n^{\frac{5}{2}}\right) = \Theta\left(n^{\frac{5}{2}}\right)$.

Let us denote $\lfloor\sqrt{n+1}\rfloor$ by $\widetilde{n}$. It follows that

$$\begin{aligned}
S_1 &= \frac{\widetilde{n}(\widetilde{n}-1)(8\widetilde{n}^3 + 3\widetilde{n}^2 - 7\widetilde{n} - 2)}{20} \\
S_2 &= \widetilde{n}^3(n - \widetilde{n}^2 + 1) + \frac{\widetilde{n}(n-\widetilde{n}^2)(n-\widetilde{n}^2+1)}{2}
\end{aligned}$$

and

$$\sum_{i=1}^{n} i\left\lfloor\sqrt{i}\right\rfloor = \frac{\widetilde{n}(\widetilde{n}-1)(8\widetilde{n}^3 + 3\widetilde{n}^2 - 7\widetilde{n} - 2)}{20} + \widetilde{n}^3(n - \widetilde{n}^2 + 1) + \frac{\widetilde{n}(n-\widetilde{n}^2)(n-\widetilde{n}^2+1)}{2} \tag{4.9}$$

and

$$\sum_{i=1}^{n} i\left\lfloor\sqrt{i}\right\rfloor = \Theta\left(n^{\frac{5}{2}}\right) \tag{4.10}$$

□

Problem 79. *Find a closed formula for*

$$\sum_{i=1}^{n} i\left\lceil\sqrt{i}\right\rceil$$

Solution:
The solution of this problem is quite similar to the solution of Problem 77. We sum the first n terms of a series, the series being the one mentioned in the solution of Problem 77 with each term multiplied by its position. Consider for example $n = 17$. The terms whose positions are perfect squares are boxed.

$$\sum_{i=1}^{17} i\left\lceil\sqrt{i}\right\rceil = \underbrace{\boxed{1}}_{\text{run 1}} + \underbrace{4+6+\boxed{8}}_{\text{run 2}} + \underbrace{15+18+21+24+\boxed{27}}_{\text{run 3}} + \underbrace{40+44+48+52+56+60+\boxed{64}}_{\text{run 4}} + \underbrace{85}_{\text{run5}}$$

Unlike Problem 77, now the runs consist of those consecutive terms whose differences are equal (and equal to the number of the run). Just as in Problem 77, all the runs but the last one are complete, the last run being either complete or incomplete. We denote the number of the complete runs with $s(n)$ and

$$s(n) = \lfloor\sqrt{n}\rfloor$$

the reasoning being exactly the same as in Problem 77. The number of terms in the incomplete run is

$$t(n) = n - \lfloor\sqrt{n}\rfloor^2$$

We break the desired sum down into two sums:

$$\sum_{i=1}^{n} i\left\lceil\sqrt{i}\right\rceil = S_1 + S_2$$

where S_1 is the sum of the terms of the complete runs and S_2, of the incomplete run.

Let us first compute S_1.

$$\begin{aligned}
S_1 &= 1.1 + 2.(2+3+4) + 3.(5+6+7+8+9) \\
&\quad + 4.(10+11+12+13+14+15+16) \\
&\quad + 5.(17+18+19+20+21+22+23+24+25) \\
&\quad + \dots \\
&\quad + s_n\big(((s_n-1)^2+1) + ((s_n-1)^2+2) + \dots + \underbrace{s_n^2}_{(s_n-1)^2+2s_n-1}\big) \\
&= \sum_{i=1}^{s_n} i \sum_{j=1}^{2i-1} (i-1)^2 + j \qquad (4.11) \\
&= \sum_{i=1}^{s_n} i \left(\sum_{j=1}^{2i-1} (i-1)^2 + \sum_{j=1}^{2i-1} j \right) \\
&= \sum_{i=1}^{s_n} i \left((i-1)^2(2i-1) + \frac{(2i-1)2i}{2} \right) \\
&= \sum_{i=1}^{s_n} i \Big((i^2-2i+1)(2i-1) + 2i^2 - i \Big) \\
&= \sum_{i=1}^{s_n} i(2i^3 - i^2 - 4i^2 + 2i + 2i - 1 + 2i^2 - i) \\
&= \sum_{i=1}^{s_n} i(2i^3 - 3i^2 + 3i - 1) \\
&= 2\sum_{i=1}^{s_n} i^4 - 3\sum_{i=1}^{s_n} i^3 + 3\sum_{i=1}^{s_n} i^2 - \sum_{i=1}^{s_n} i \quad \text{apply (4.21), (4.22), (4.23), and (4.24)} \\
&= 2\frac{s_n(s_n+1)(2s_n+1)(3s_n^2+3s_n-1)}{30} - 3\frac{s_n^2(s_n+1)^2}{4} + \\
&\qquad 3\frac{s_n(s_n+1)(2s_n+1)}{6} - \frac{s_n(s_n+1)}{2} \\
&= \frac{s_n(s_n+1)}{2}\left(\frac{2(2s_n+1)(3s_n^2+3s_n-1)}{15} - \frac{3s_n(s_n+1)}{2} + \frac{6s_n+3}{3} - 1 \right) \qquad (4.12)
\end{aligned}$$

Simplify (4.12) to obtain

$$\frac{s_n(s_n+1)}{2}\left(\frac{12s_n^3+12s_n^2-4s_n+6s_n^2+6s_n-2}{15}-\frac{3s_n^2+3s_n}{2}+\frac{6s_n+3}{3}-1\right)=$$
$$\frac{s_n(s_n+1)}{2}\left(\frac{24s_n^3+36s_n^2+4s_n-4}{30}-\frac{45s_n^2+45s_n}{30}+\frac{60s_n+30}{30}-\frac{30}{30}\right)=$$
$$\frac{s_n(s_n+1)}{60}(24s_n^3+36s_n^2+4s_n-4-45s_n^2-45s_n+60s_n+30-30)=$$
$$\frac{s_n(s_n+1)(24s_n^3-9s_n^2+19s_n-4)}{60}=$$
$$\frac{\lfloor\sqrt{n}\rfloor(\lfloor\sqrt{n}\rfloor+1)(24\lfloor\sqrt{n}\rfloor^3-9\lfloor\sqrt{n}\rfloor^2+19\lfloor\sqrt{n}\rfloor-4)}{60}$$

Clearly, $S_1=\Theta\left(n^{\frac{5}{2}}\right)$. Now we compute S_2. It has t_n terms, the first term is $(s_n^2+1)(s_n+1)$, and the difference between every two consecutive terms is (s_n+1).

$$\begin{aligned}
S_2 &= \sum_{i=1}^{t_n}(s_n^2+1)(s_n+1)+(i-1)(s_n+1)= \\
&= (s_n^2+1)(s_n+1)\sum_{i=1}^{t_n}1+(s_n+1)\sum_{i=1}^{t_n}(i-1) \\
&= t_n(s_n^2+1)(s_n+1)+\frac{(s_n+1)(t_n-1)t_n}{2}= \\
&= \frac{t_n(s_n+1)}{2}\left(2s_n^2+2+t_n-1\right)= \\
&= \frac{t_n(s_n+1)(2s_n^2+t_n+1)}{2} \\
&= \frac{(n-\lfloor\sqrt{n}\rfloor^2)(\lfloor\sqrt{n}\rfloor+1)(2\lfloor\sqrt{n}\rfloor^2+n-\lfloor\sqrt{n}\rfloor^2+1)}{2} \\
&= \frac{(n-\lfloor\sqrt{n}\rfloor^2)(\lfloor\sqrt{n}\rfloor+1)(n+\lfloor\sqrt{n}\rfloor^2+1)}{2}
\end{aligned}$$

Clearly, $S_2=O\left(n^{\frac{5}{2}}\right)$, therefore $S_1+S_2=\Theta\left(n^{\frac{5}{2}}\right)+O\left(n^{\frac{5}{2}}\right)=\Theta\left(n^{\frac{5}{2}}\right)$. It follows that

$$\sum_{i=1}^{n} i\left\lfloor\sqrt{i}\right\rfloor=\frac{\lfloor\sqrt{n}\rfloor(\lfloor\sqrt{n}\rfloor+1)(24\lfloor\sqrt{n}\rfloor^3-9\lfloor\sqrt{n}\rfloor^2+19\lfloor\sqrt{n}\rfloor-4)}{60}+\frac{(n-\lfloor\sqrt{n}\rfloor^2)(\lfloor\sqrt{n}\rfloor+1)(n+\lfloor\sqrt{n}\rfloor^2+1)}{2} \tag{4.13}$$

and

$$\sum_{i=1}^{n} i\left\lceil\sqrt{i}\right\rceil=\Theta\left(n^{\frac{5}{2}}\right) \tag{4.14}$$

□

Fact: The Fibonacci numbers are the natural numbers defined by the recurrence relation

$$F_0 = 0$$
$$F_1 = 1$$
$$F_n = F_{n-1} + F_{n-2}, \text{ for all } n > 1$$

The first several elements of the sequence are

$$0, 1, 1, 2, 3, 5, 8, 13, 21, \ldots$$

The asymptotic growth rate of F_n is determined by the following equality [GKP94, pp. 300]

$$F_n = \left\lfloor \frac{\phi^n}{\sqrt{5}} + \frac{1}{2} \right\rfloor = \frac{\phi^n}{\sqrt{5}}, \text{ rounded to the nearest integer}$$

where $\phi = \frac{1+\sqrt{5}}{2}$ is the so called "golden ratio", the positive root of $\phi^2 = \phi + 1$. Clearly, for any positive constant c,

$$c^{F_n} = \Theta\left(c^{\frac{\phi^n}{\sqrt{5}}}\right) = \Theta\left(k^{\phi^n}\right), \text{ where } k = c^{\frac{1}{\sqrt{5}}} \tag{4.15}$$

□

Fact: The harmonic series

$$1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \ldots = \sum_{i=1}^{\infty} \frac{1}{i}$$

is divergent. Its n^{th} partial sum is denoted by H_n.

$$H_n = \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n-1} + \frac{1}{n} \tag{4.16}$$

It is known that

$$H_n = \Theta(\lg n) \tag{4.17}$$

Furthermore, $\ln n < H_n < \ln n + 1$ for $n > 1$. For details, see [GKP94, pp. 272–278]. □

Fact: The sum of the first k^{th} powers for some integer constant $k \geq 1$ is

$$1^k + 2^k + \ldots + n^k = \sum_{i=0}^{n} i^k \tag{4.18}$$

It is well known that

$$\sum_{i=0}^{n} i^k = \frac{1}{k+1} \sum_{j=0}^{k} \binom{k+1}{j} B_j (n+1)^{k+1-j} \tag{4.19}$$

where B_j is the j^{th} *Bernoulli number*. The Bernolli numbers are defined with the recurrence

$$B_0 = 1$$
$$B_m = -\frac{1}{m}\sum_{j=0}^{m-1}\binom{m+1}{j}B_j, \text{ for } m \in \mathbb{N}^+$$

For details on the summation formula (4.19) and plenty of information on the Bernoulli numbers, see [GKP94, pp. 283–290]. Just keep in mind that Knuth *et al.* denote the sum by $S_k(n)$ and define it as

$$S_k(n) = 0^k + 1^k + 2^k + \ldots + (n-1)^k$$

For our purposes in this manual it is sufficient to know that

$$1^k + 2^k + \ldots + n^k = \Theta(n^{k+1}) \tag{4.20}$$

which fact follows easily from (4.19). In fact, (4.19) is a polynomial of degree $k+1$ of n because the $\binom{k+1}{j}$ factor and the Bernoulli numbers are just constants and clearly the highest degree of n is $k+1$. Strictly speaking, we have not proved here formally that (4.19) is a degree $k+1$ polynomial of n because we have not shown that the coefficient before n^{k+1} is not zero. But that is indeed the case—see for instance [GKP94, (6.98), pp. 288].

⁘ NB ⁘ Be careful to avoid the error of thinking that

$$1^k + 2^k + \ldots + n^k$$

is a degree k polynomial of n and thus erroneosly concluding that its order of growth is $\Theta(n^k)$. It is *not* a polynomial of n because a polynomial has an *a priori* fixed number of terms, while the above sum has n terms where n is the variable.

Using (4.19), we can easily derive

$$1 + 2 + \ldots + n = \frac{n(n+1)}{2} \tag{4.21}$$
$$1^2 + 2^2 + \ldots + n^2 = \frac{n(n+1)(2n+1)}{6} \tag{4.22}$$
$$1^3 + 2^3 + \ldots + n^3 = \frac{n^2(n+1)^2}{4} \tag{4.23}$$
$$1^4 + 2^4 + \ldots + n^4 = \frac{n(n+1)(2n+1)(3n^2+3n-1)}{30} \tag{4.24}$$

□

Bibliography

[AB98] Mohamad Akra and Louay Bazzi. On the solution of linear recurrence equations. *Computational Optimization and Applications*, 10(2):195–210, 1998.

[CLR00] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. McGraw-Hill Book Company, first edition, 2000.

[GKP94] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics*. Addison-Wesley, second edition, 1994.

[KMP77] D. E. Knuth, J. H. Morris, and V. R. Pratt. Fast pattern matching in strings. *SIAM Journal on Computing*, 6:323–350, 1977.

[Knu73] Donald E. Knuth. *The Art of Computer Programming*, volume 1. Addison-Wesley Publishing Company, second edition, 1973.

[Lei96] Leighton. Note on Better Master Theorems for Divide-and-Conquer Recurrences, 1996. Available online at `http://courses.csail.mit.edu/6.046/spring04/handouts/akrabazzi.pdf`.

[Slo] N. J. Sloane. The on-line encyclopedia of integer sequences. maintained by N. J. A. Sloane `njas@research.att.com`, available at `http://www.research.att.com/~njas/sequences/`.