Suppose that we use the following greedy algorithm for parallel machine scheduling: whenever a machine is idle, schedule any job that has not yet been scheduled.

***c.*** Write pseudocode to implement this greedy algorithm. What is the running time of your algorithm?

***d.*** For the schedule returned by the greedy algorithm, show that

$$C_{\max} \le \frac{1}{m} \sum_{1 \le k \le n} p_k + \max_{1 \le k \le n} p_k .$$

Conclude that this algorithm is a polynomial-time 2-approximation algorithm.

### *35-6 Approximating a maximum spanning tree*

Let $G = (V, E)$ be an undirected graph with distinct edge weights $w(u, v)$ on each edge $(u, v) \in E$. For each vertex $v \in V$, let $\max(v) = \max_{(u,v) \in E} \{w(u, v)\}$ be the maximum-weight edge incident on that vertex. Let $S_G = \{\max(v) : v \in V\}$ be the set of maximum-weight edges incident on each vertex, and let $T_G$ be the maximum-weight spanning tree of $G$, that is, the spanning tree of maximum total weight. For any subset of edges $E' \subseteq E$, define $w(E') = \sum_{(u,v) \in E'} w(u, v)$.

***a.*** Give an example of a graph with at least 4 vertices for which $S_G = T_G$.

***b.*** Give an example of a graph with at least 4 vertices for which $S_G \ne T_G$.

***c.*** Prove that $S_G \subseteq T_G$ for any graph $G$.

***d.*** Prove that $w(T_G) \ge w(S_G)/2$ for any graph $G$.

***e.*** Give an $O(V + E)$-time algorithm to compute a 2-approximation to the maximum spanning tree.

### *35-7 An approximation algorithm for the 0-1 knapsack problem*

Recall the knapsack problem from Section 16.2. There are $n$ items, where the $i$th item is worth $v_i$ dollars and weighs $w_i$ pounds. We are also given a knapsack that can hold at most $W$ pounds. Here, we add the further assumptions that each weight $w_i$ is at most $W$ and that the items are indexed in monotonically decreasing order of their values: $v_1 \ge v_2 \ge \cdots \ge v_n$.

In the 0-1 knapsack problem, we wish to find a subset of the items whose total weight is at most $W$ and whose total value is maximum. The fractional knapsack problem is like the 0-1 knapsack problem, except that we are allowed to take a fraction of each item, rather than being restricted to taking either all or none of each item. If we take a fraction $x_i$ of item $i$, where $0 \le x_i \le 1$, we contribute $x_i w_i$ to the weight of the knapsack and receive value $x_i v_i$. Our goal is to develop a polynomial-time 2-approximation algorithm for the 0-1 knapsack problem.

In order to design a polynomial-time algorithm, we consider restricted instances of the 0-1 knapsack problem. Given an instance $I$ of the knapsack problem, we form restricted instances $I_j$, for $j = 1, 2, \ldots, n$, by removing items $1, 2, \ldots, j-1$ and requiring the solution to include item $j$ (all of item $j$ in both the fractional and 0-1 knapsack problems). No items are removed in instance $I_1$. For instance $I_j$, let $P_j$ denote an optimal solution to the 0-1 problem and $Q_j$ denote an optimal solution to the fractional problem.

***a.*** Argue that an optimal solution to instance $I$ of the 0-1 knapsack problem is one of $\{P_1, P_2, \ldots, P_n\}$.

***b.*** Prove that we can find an optimal solution $Q_j$ to the fractional problem for instance $I_j$ by including item $j$ and then using the greedy algorithm in which at each step, we take as much as possible of the unchosen item in the set $\{j+1, j+2, \ldots, n\}$ with maximum value per pound $v_i/w_i$.

***c.*** Prove that we can always construct an optimal solution $Q_j$ to the fractional problem for instance $I_j$ that includes at most one item fractionally. That is, for all items except possibly one, we either include all of the item or none of the item in the knapsack.

***d.*** Given an optimal solution $Q_j$ to the fractional problem for instance $I_j$, form solution $R_j$ from $Q_j$ by deleting any fractional items from $Q_j$. Let $v(S)$ denote the total value of items taken in a solution $S$. Prove that $v(R_j) \ge v(Q_j)/2 \ge v(P_j)/2$.

***e.*** Give a polynomial-time algorithm that returns a maximum-value solution from the set $\{R_1, R_2, \ldots, R_n\}$, and prove that your algorithm is a polynomial-time 2-approximation algorithm for the 0-1 knapsack problem.

## Chapter notes

Although methods that do not necessarily compute exact solutions have been known for thousands of years (for example, methods to approximate the value of $\pi$), the notion of an approximation algorithm is much more recent. Hochbaum [172] credits Garey, Graham, and Ullman [128] and Johnson [190] with formalizing the concept of a polynomial-time approximation algorithm. The first such algorithm is often credited to Graham [149].

Since this early work, thousands of approximation algorithms have been designed for a wide range of problems, and there is a wealth of literature on this field. Recent texts by Ausiello et al. [26], Hochbaum [172], and Vazirani [345] deal exclusively with approximation algorithms, as do surveys by Shmoys [315] and Klein and Young [207]. Several other texts, such as Garey and Johnson [129] and Papadimitriou and Steiglitz [271], have significant coverage of approximation algorithms as well. Lawler, Lenstra, Rinnooy Kan, and Shmoys [225] provide an extensive treatment of approximation algorithms for the traveling-salesman problem.

Papadimitriou and Steiglitz attribute the algorithm APPROX-VERTEX-COVER to F. Gavril and M. Yannakakis. The vertex-cover problem has been studied extensively (Hochbaum [172] lists 16 different approximation algorithms for this problem), but all the approximation ratios are at least $2 - o(1)$.

The algorithm APPROX-TSP-TOUR appears in a paper by Rosenkrantz, Stearns, and Lewis [298]. Christofides improved on this algorithm and gave a $3/2$-approximation algorithm for the traveling-salesman problem with the triangle inequality. Arora [22] and Mitchell [257] have shown that if the points are in the euclidean plane, there is a polynomial-time approximation scheme. Theorem 35.3 is due to Sahni and Gonzalez [301].

The analysis of the greedy heuristic for the set-covering problem is modeled after the proof published by Chvátal [68] of a more general result; the basic result as presented here is due to Johnson [190] and Lovász [238].

The algorithm APPROX-SUBSET-SUM and its analysis are loosely modeled after related approximation algorithms for the knapsack and subset-sum problems by Ibarra and Kim [187].

Problem 35-7 is a combinatorial version of a more general result on approximating knapsack-type integer programs by Bienstock and McClosky [45].

The randomized algorithm for MAX-3-CNF satisfiability is implicit in the work of Johnson [190]. The weighted vertex-cover algorithm is by Hochbaum [171]. Section 35.4 only touches on the power of randomization and linear programming in the design of approximation algorithms. A combination of these two ideas yields a technique called "randomized rounding," which formulates a problem as an integer linear program, solves the linear-programming relaxation, and interprets the variables in the solution as probabilities. These probabilities then help guide the solution of the original problem. This technique was first used by Raghavan and Thompson [290], and it has had many subsequent uses. (See Motwani, Naor, and Raghavan [261] for a survey.) Several other notable recent ideas in the field of approximation algorithms include the primal-dual method (see Goemans and Williamson [135] for a survey), finding sparse cuts for use in divide-and-conquer algorithms [229], and the use of semidefinite programming [134].

As mentioned in the chapter notes for Chapter 34, recent results in probabilistically checkable proofs have led to lower bounds on the approximability of many problems, including several in this chapter. In addition to the references there, the chapter by Arora and Lund [23] contains a good description of the relationship between probabilistically checkable proofs and the hardness of approximating various problems.

# VIII Appendix: Mathematical Background

## Introduction

When we analyze algorithms, we often need to draw upon a body of mathematical tools. Some of these tools are as simple as high-school algebra, but others may be new to you. In Part I, we saw how to manipulate asymptotic notations and solve recurrences. This appendix comprises a compendium of several other concepts and methods we use to analyze algorithms. As noted in the introduction to Part I, you may have seen much of the material in this appendix before having read this book (although the specific notational conventions we use might occasionally differ from those you have seen elsewhere). Hence, you should treat this appendix as reference material. As in the rest of this book, however, we have included exercises and problems, in order for you to improve your skills in these areas.

Appendix A offers methods for evaluating and bounding summations, which occur frequently in the analysis of algorithms. Many of the formulas here appear in any calculus text, but you will find it convenient to have these methods compiled in one place.

Appendix B contains basic definitions and notations for sets, relations, functions, graphs, and trees. It also gives some basic properties of these mathematical objects.

Appendix C begins with elementary principles of counting: permutations, combinations, and the like. The remainder contains definitions and properties of basic probability. Most of the algorithms in this book require no probability for their analysis, and thus you can easily omit the latter sections of the chapter on a first reading, even without skimming them. Later, when you encounter a probabilistic analysis that you want to understand better, you will find Appendix C well organized for reference purposes.

Appendix D defines matrices, their operations, and some of their basic properties. You have probably seen most of this material already if you have taken a course in linear algebra, but you might find it helpful to have one place to look for our notation and definitions.

# A Summations

When an algorithm contains an iterative control construct such as a **while** or **for** loop, we can express its running time as the sum of the times spent on each execution of the body of the loop. For example, we found in Section 2.2 that the $j$th iteration of insertion sort took time proportional to $j$ in the worst case. By adding up the time spent on each iteration, we obtained the summation (or series)

$$\sum_{j=2}^{n} j \ .$$

When we evaluated this summation, we attained a bound of $\Theta(n^2)$ on the worst-case running time of the algorithm. This example illustrates why you should know how to manipulate and bound summations.

Section A.1 lists several basic formulas involving summations. Section A.2 offers useful techniques for bounding summations. We present the formulas in Section A.1 without proof, though proofs for some of them appear in Section A.2 to illustrate the methods of that section. You can find most of the other proofs in any calculus text.

## A.1 Summation formulas and properties

Given a sequence $a_1, a_2, \ldots, a_n$ of numbers, where $n$ is a nonnegative integer, we can write the finite sum $a_1 + a_2 + \cdots + a_n$ as

$$\sum_{k=1}^{n} a_k \ .$$

If $n = 0$, the value of the summation is defined to be 0. The value of a finite series is always well defined, and we can add its terms in any order.

Given an infinite sequence $a_1, a_2, \ldots$ of numbers, we can write the infinite sum $a_1 + a_2 + \cdots$ as

$$\sum_{k=1}^{\infty} a_k ,$$

which we interpret to mean

$$\lim_{n \to \infty} \sum_{k=1}^{n} a_k .$$

If the limit does not exist, the series ***diverges***; otherwise, it ***converges***. The terms of a convergent series cannot always be added in any order. We can, however, rearrange the terms of an ***absolutely convergent series***, that is, a series $\sum_{k=1}^{\infty} a_k$ for which the series $\sum_{k=1}^{\infty} |a_k|$ also converges.

### Linearity

For any real number $c$ and any finite sequences $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_n$,

$$\sum_{k=1}^{n} (ca_k + b_k) = c \sum_{k=1}^{n} a_k + \sum_{k=1}^{n} b_k .$$

The linearity property also applies to infinite convergent series.

We can exploit the linearity property to manipulate summations incorporating asymptotic notation. For example,

$$\sum_{k=1}^{n} \Theta(f(k)) = \Theta\left(\sum_{k=1}^{n} f(k)\right) .$$

In this equation, the $\Theta$-notation on the left-hand side applies to the variable $k$, but on the right-hand side, it applies to $n$. We can also apply such manipulations to infinite convergent series.

### Arithmetic series

The summation

$$\sum_{k=1}^{n} k = 1 + 2 + \cdots + n ,$$

is an ***arithmetic series*** and has the value

$$\sum_{k=1}^{n} k = \frac{1}{2} n(n+1) \tag{A.1}$$

$$= \Theta(n^2) . \tag{A.2}$$

### Sums of squares and cubes

We have the following summations of squares and cubes:

$$\sum_{k=0}^{n} k^2 = \frac{n(n+1)(2n+1)}{6} , \tag{A.3}$$

$$\sum_{k=0}^{n} k^3 = \frac{n^2(n+1)^2}{4} . \tag{A.4}$$

### Geometric series

For real $x \neq 1$, the summation

$$\sum_{k=0}^{n} x^k = 1 + x + x^2 + \cdots + x^n$$

is a ***geometric*** or ***exponential series*** and has the value

$$\sum_{k=0}^{n} x^k = \frac{x^{n+1} - 1}{x - 1} . \tag{A.5}$$

When the summation is infinite and $|x| < 1$, we have the infinite decreasing geometric series

$$\sum_{k=0}^{\infty} x^k = \frac{1}{1-x} . \tag{A.6}$$

### Harmonic series

For positive integers $n$, the $n$th ***harmonic number*** is

$$\begin{aligned} H_n &= 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots + \frac{1}{n} \\ &= \sum_{k=1}^{n} \frac{1}{k} \\ &= \ln n + O(1) . \end{aligned} \tag{A.7}$$

(We shall prove a related bound in Section A.2.)

### Integrating and differentiating series

By integrating or differentiating the formulas above, additional formulas arise. For example, by differentiating both sides of the infinite geometric series (A.6) and multiplying by $x$, we get

$$\sum_{k=0}^{\infty} k x^k = \frac{x}{(1-x)^2} \tag{A.8}$$

for $|x| < 1$.

### Telescoping series

For any sequence $a_0, a_1, \ldots, a_n$,

$$\sum_{k=1}^{n} (a_k - a_{k-1}) = a_n - a_0 , \tag{A.9}$$

since each of the terms $a_1, a_2, \ldots, a_{n-1}$ is added in exactly once and subtracted out exactly once. We say that the sum ***telescopes***. Similarly,

$$\sum_{k=0}^{n-1} (a_k - a_{k+1}) = a_0 - a_n .$$

As an example of a telescoping sum, consider the series

$$\sum_{k=1}^{n-1} \frac{1}{k(k+1)} .$$

Since we can rewrite each term as

$$\frac{1}{k(k+1)} = \frac{1}{k} - \frac{1}{k+1} ,$$

we get

$$\begin{aligned} \sum_{k=1}^{n-1} \frac{1}{k(k+1)} &= \sum_{k=1}^{n-1} \left( \frac{1}{k} - \frac{1}{k+1} \right) \\ &= 1 - \frac{1}{n} . \end{aligned}$$

### Products

We can write the finite product $a_1 a_2 \cdots a_n$ as

$$\prod_{k=1}^{n} a_k .$$

If $n = 0$, the value of the product is defined to be 1. We can convert a formula with a product to a formula with a summation by using the identity

$$\lg \left( \prod_{k=1}^{n} a_k \right) = \sum_{k=1}^{n} \lg a_k .$$

### Exercises

***A.1-1***
Find a simple formula for $\sum_{k=1}^{n}(2k-1)$.

***A.1-2*** $\star$
Show that $\sum_{k=1}^{n} 1/(2k-1) = \ln(\sqrt{n}) + O(1)$ by manipulating the harmonic series.

***A.1-3***
Show that $\sum_{k=0}^{\infty} k^2 x^k = x(1+x)/(1-x)^3$ for $0 < |x| < 1$.

***A.1-4*** $\star$
Show that $\sum_{k=0}^{\infty}(k-1)/2^k = 0$.

***A.1-5*** $\star$
Evaluate the sum $\sum_{k=1}^{\infty}(2k+1)x^{2k}$.

***A.1-6***
Prove that $\sum_{k=1}^{n} O(f_k(i)) = O\left(\sum_{k=1}^{n} f_k(i)\right)$ by using the linearity property of summations.

***A.1-7***
Evaluate the product $\prod_{k=1}^{n} 2 \cdot 4^k$.

***A.1-8*** $\star$
Evaluate the product $\prod_{k=2}^{n}(1 - 1/k^2)$.

## A.2 Bounding summations

We have many techniques at our disposal for bounding the summations that describe the running times of algorithms. Here are some of the most frequently used methods.

### Mathematical induction

The most basic way to evaluate a series is to use mathematical induction. As an example, let us prove that the arithmetic series $\sum_{k=1}^{n} k$ evaluates to $\frac{1}{2}n(n+1)$. We can easily verify this assertion for $n = 1$. We make the inductive assumption that it holds for $n$, and we prove that it holds for $n+1$. We have

$$\begin{aligned} \sum_{k=1}^{n+1} k &= \sum_{k=1}^{n} k + (n+1) \\ &= \frac{1}{2}n(n+1) + (n+1) \\ &= \frac{1}{2}(n+1)(n+2) . \end{aligned}$$

You don't always need to guess the exact value of a summation in order to use mathematical induction. Instead, you can use induction to prove a bound on a summation. As an example, let us prove that the geometric series $\sum_{k=0}^{n} 3^k$ is $O(3^n)$. More specifically, let us prove that $\sum_{k=0}^{n} 3^k \le c3^n$ for some constant $c$. For the initial condition $n = 0$, we have $\sum_{k=0}^{0} 3^k = 1 \le c \cdot 1$ as long as $c \ge 1$. Assuming that the bound holds for $n$, let us prove that it holds for $n+1$. We have

$$\begin{aligned} \sum_{k=0}^{n+1} 3^k &= \sum_{k=0}^{n} 3^k + 3^{n+1} \\ &\le c3^n + 3^{n+1} && \text{(by the inductive hypothesis)} \\ &= \left(\frac{1}{3} + \frac{1}{c}\right) c3^{n+1} \\ &\le c3^{n+1} \end{aligned}$$

as long as $(1/3 + 1/c) \le 1$ or, equivalently, $c \ge 3/2$. Thus, $\sum_{k=0}^{n} 3^k = O(3^n)$, as we wished to show.

We have to be careful when we use asymptotic notation to prove bounds by induction. Consider the following fallacious proof that $\sum_{k=1}^{n} k = O(n)$. Certainly, $\sum_{k=1}^{1} k = O(1)$. Assuming that the bound holds for $n$, we now prove it for $n+1$:

$$\begin{aligned} \sum_{k=1}^{n+1} k &= \sum_{k=1}^{n} k + (n+1) \\ &= O(n) + (n+1) && \Longleftarrow \textit{wrong!!} \\ &= O(n+1) . \end{aligned}$$

The bug in the argument is that the "constant" hidden by the "big-oh" grows with $n$ and thus is not constant. We have not shown that the same constant works for *all* $n$.

### Bounding the terms

We can sometimes obtain a good upper bound on a series by bounding each term of the series, and it often suffices to use the largest term to bound the others. For example, a quick upper bound on the arithmetic series (A.1) is

$$\begin{aligned}\sum_{k=1}^{n} k &\le \sum_{k=1}^{n} n \\ &= n^2 .\end{aligned}$$

In general, for a series $\sum_{k=1}^{n} a_k$, if we let $a_{\max} = \max_{1 \le k \le n} a_k$, then

$$\sum_{k=1}^{n} a_k \le n \cdot a_{\max} .$$

The technique of bounding each term in a series by the largest term is a weak method when the series can in fact be bounded by a geometric series. Given the series $\sum_{k=0}^{n} a_k$, suppose that $a_{k+1}/a_k \le r$ for all $k \ge 0$, where $0 < r < 1$ is a constant. We can bound the sum by an infinite decreasing geometric series, since $a_k \le a_0 r^k$, and thus

$$\begin{aligned}\sum_{k=0}^{n} a_k &\le \sum_{k=0}^{\infty} a_0 r^k \\ &= a_0 \sum_{k=0}^{\infty} r^k \\ &= a_0 \frac{1}{1-r} .\end{aligned}$$

We can apply this method to bound the summation $\sum_{k=1}^{\infty} (k/3^k)$. In order to start the summation at $k = 0$, we rewrite it as $\sum_{k=0}^{\infty} ((k+1)/3^{k+1})$. The first term ($a_0$) is $1/3$, and the ratio ($r$) of consecutive terms is

$$\begin{aligned}\frac{(k+2)/3^{k+2}}{(k+1)/3^{k+1}} &= \frac{1}{3} \cdot \frac{k+2}{k+1} \\ &\le \frac{2}{3}\end{aligned}$$

for all $k \ge 0$. Thus, we have

$$\begin{aligned}\sum_{k=1}^{\infty} \frac{k}{3^k} &= \sum_{k=0}^{\infty} \frac{k+1}{3^{k+1}} \\ &\le \frac{1}{3} \cdot \frac{1}{1-2/3} \\ &= 1 .\end{aligned}$$

A common bug in applying this method is to show that the ratio of consecutive terms is less than 1 and then to assume that the summation is bounded by a geometric series. An example is the infinite harmonic series, which diverges since

$$\begin{aligned} \sum_{k=1}^{\infty} \frac{1}{k} &= \lim_{n\to\infty} \sum_{k=1}^{n} \frac{1}{k} \\ &= \lim_{n\to\infty} \Theta(\lg n) \\ &= \infty \, . \end{aligned}$$

The ratio of the $(k+1)$st and $k$th terms in this series is $k/(k+1) < 1$, but the series is not bounded by a decreasing geometric series. To bound a series by a geometric series, we must show that there is an $r < 1$, which is a *constant*, such that the ratio of all pairs of consecutive terms never exceeds $r$. In the harmonic series, no such $r$ exists because the ratio becomes arbitrarily close to 1.

### Splitting summations

One way to obtain bounds on a difficult summation is to express the series as the sum of two or more series by partitioning the range of the index and then to bound each of the resulting series. For example, suppose we try to find a lower bound on the arithmetic series $\sum_{k=1}^{n} k$, which we have already seen has an upper bound of $n^2$. We might attempt to bound each term in the summation by the smallest term, but since that term is 1, we get a lower bound of $n$ for the summation—far off from our upper bound of $n^2$.

We can obtain a better lower bound by first splitting the summation. Assume for convenience that $n$ is even. We have

$$\begin{aligned} \sum_{k=1}^{n} k &= \sum_{k=1}^{n/2} k + \sum_{k=n/2+1}^{n} k \\ &\geq \sum_{k=1}^{n/2} 0 + \sum_{k=n/2+1}^{n} (n/2) \\ &= (n/2)^2 \\ &= \Omega(n^2) \, , \end{aligned}$$

which is an asymptotically tight bound, since $\sum_{k=1}^{n} k = O(n^2)$.

For a summation arising from the analysis of an algorithm, we can often split the summation and ignore a constant number of the initial terms. Generally, this technique applies when each term $a_k$ in a summation $\sum_{k=0}^{n} a_k$ is independent of $n$.

Then for any constant $k_0 > 0$, we can write

$$\begin{aligned}\sum_{k=0}^{n} a_k &= \sum_{k=0}^{k_0-1} a_k + \sum_{k=k_0}^{n} a_k \\ &= \Theta(1) + \sum_{k=k_0}^{n} a_k ,\end{aligned}$$

since the initial terms of the summation are all constant and there are a constant number of them. We can then use other methods to bound $\sum_{k=k_0}^{n} a_k$. This technique applies to infinite summations as well. For example, to find an asymptotic upper bound on

$$\sum_{k=0}^{\infty} \frac{k^2}{2^k} ,$$

we observe that the ratio of consecutive terms is

$$\begin{aligned}\frac{(k+1)^2/2^{k+1}}{k^2/2^k} &= \frac{(k+1)^2}{2k^2} \\ &\le \frac{8}{9}\end{aligned}$$

if $k \ge 3$. Thus, the summation can be split into

$$\begin{aligned}\sum_{k=0}^{\infty} \frac{k^2}{2^k} &= \sum_{k=0}^{2} \frac{k^2}{2^k} + \sum_{k=3}^{\infty} \frac{k^2}{2^k} \\ &\le \sum_{k=0}^{2} \frac{k^2}{2^k} + \frac{9}{8}\sum_{k=0}^{\infty} \left(\frac{8}{9}\right)^k \\ &= O(1) ,\end{aligned}$$

since the first summation has a constant number of terms and the second summation is a decreasing geometric series.

The technique of splitting summations can help us determine asymptotic bounds in much more difficult situations. For example, we can obtain a bound of $O(\lg n)$ on the harmonic series (A.7):

$$H_n = \sum_{k=1}^{n} \frac{1}{k} .$$

We do so by splitting the range 1 to $n$ into $\lfloor \lg n \rfloor + 1$ pieces and upper-bounding the contribution of each piece by 1. For $i = 0, 1, \ldots, \lfloor \lg n \rfloor$, the $i$th piece consists of the terms starting at $1/2^i$ and going up to but not including $1/2^{i+1}$. The last piece might contain terms not in the original harmonic series, and thus we have

$$\begin{aligned}
\sum_{k=1}^{n} \frac{1}{k} &\le \sum_{i=0}^{\lfloor \lg n \rfloor} \sum_{j=0}^{2^i-1} \frac{1}{2^i + j} \\
&\le \sum_{i=0}^{\lfloor \lg n \rfloor} \sum_{j=0}^{2^i-1} \frac{1}{2^i} \\
&= \sum_{i=0}^{\lfloor \lg n \rfloor} 1 \\
&\le \lg n + 1 . \qquad \text{(A.10)}
\end{aligned}$$

### Approximation by integrals

When a summation has the form $\sum_{k=m}^{n} f(k)$, where $f(k)$ is a monotonically increasing function, we can approximate it by integrals:

$$\int_{m-1}^{n} f(x)\,dx \le \sum_{k=m}^{n} f(k) \le \int_{m}^{n+1} f(x)\,dx . \qquad \text{(A.11)}$$

Figure A.1 justifies this approximation. The summation is represented as the area of the rectangles in the figure, and the integral is the shaded region under the curve. When $f(k)$ is a monotonically decreasing function, we can use a similar method to provide the bounds

$$\int_{m}^{n+1} f(x)\,dx \le \sum_{k=m}^{n} f(k) \le \int_{m-1}^{n} f(x)\,dx . \qquad \text{(A.12)}$$

The integral approximation (A.12) gives a tight estimate for the $n$th harmonic number. For a lower bound, we obtain

$$\begin{aligned}
\sum_{k=1}^{n} \frac{1}{k} &\ge \int_{1}^{n+1} \frac{dx}{x} \\
&= \ln(n+1) . \qquad \text{(A.13)}
\end{aligned}$$

For the upper bound, we derive the inequality

$$\begin{aligned}
\sum_{k=2}^{n} \frac{1}{k} &\le \int_{1}^{n} \frac{dx}{x} \\
&= \ln n ,
\end{aligned}$$

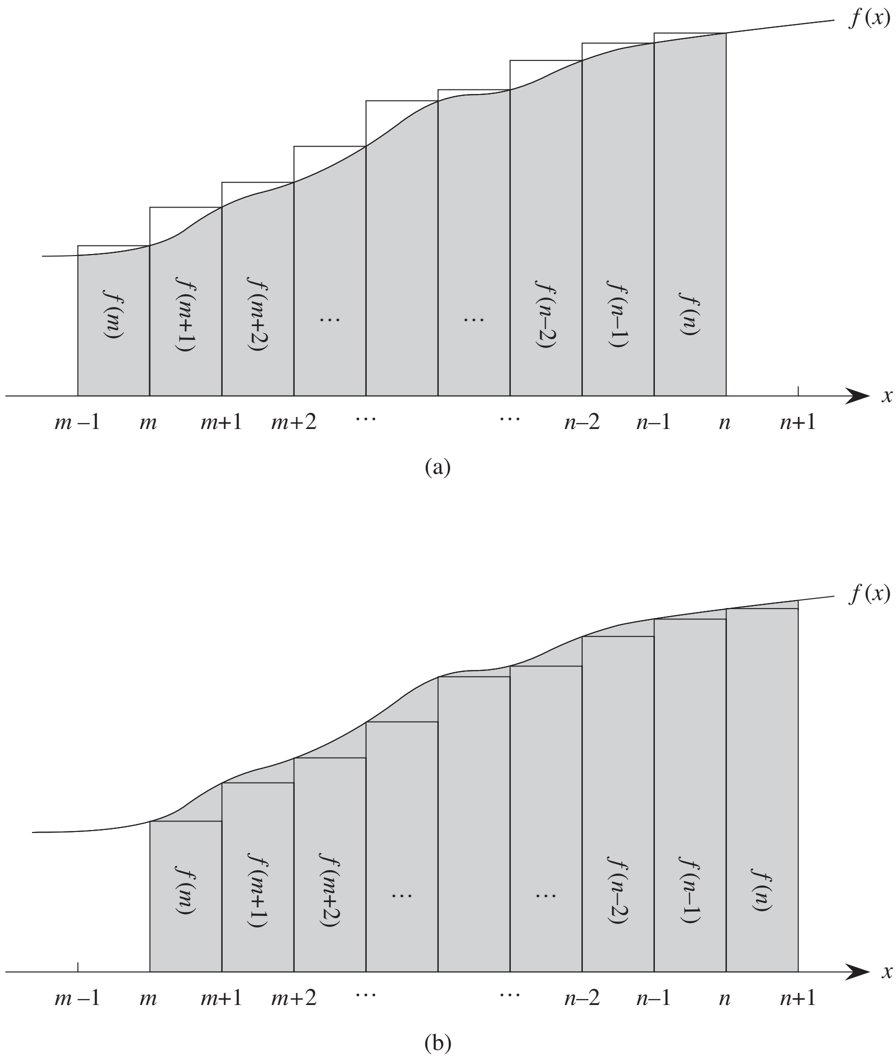


**Figure A.1** Approximation of $\sum_{k=m}^{n} f(k)$ by integrals. The area of each rectangle is shown within the rectangle, and the total rectangle area represents the value of the summation. The integral is represented by the shaded area under the curve. By comparing areas in **(a)**, we get $\int_{m-1}^{n} f(x)\,dx \le \sum_{k=m}^{n} f(k)$, and then by shifting the rectangles one unit to the right, we get $\sum_{k=m}^{n} f(k) \le \int_{m}^{n+1} f(x)\,dx$ in **(b)**.

which yields the bound

$$\sum_{k=1}^{n} \frac{1}{k} \le \ln n + 1 . \tag{A.14}$$

### Exercises

***A.2-1***
Show that $\sum_{k=1}^{n} 1/k^2$ is bounded above by a constant.

***A.2-2***
Find an asymptotic upper bound on the summation

$$\sum_{k=0}^{\lfloor \lg n \rfloor} \lceil n/2^k \rceil .$$

***A.2-3***
Show that the $n$th harmonic number is $\Omega(\lg n)$ by splitting the summation.

***A.2-4***
Approximate $\sum_{k=1}^{n} k^3$ with an integral.

***A.2-5***
Why didn't we use the integral approximation (A.12) directly on $\sum_{k=1}^{n} 1/k$ to obtain an upper bound on the $n$th harmonic number?

## Problems

***A-1 Bounding summations***
Give asymptotically tight bounds on the following summations. Assume that $r \ge 0$ and $s \ge 0$ are constants.

***a.*** $\displaystyle\sum_{k=1}^{n} k^r .$

***b.*** $\displaystyle\sum_{k=1}^{n} \lg^s k .$

***c.*** $\displaystyle\sum_{k=1}^{n} k^r \lg^s k$.

## Appendix notes

Knuth [209] provides an excellent reference for the material presented here. You can find basic properties of series in any good calculus book, such as Apostol [18] or Thomas et al. [334].

# B Sets, Etc.

Many chapters of this book touch on the elements of discrete mathematics. This appendix reviews more completely the notations, definitions, and elementary properties of sets, relations, functions, graphs, and trees. If you are already well versed in this material, you can probably just skim this chapter.

## B.1 Sets

A ***set*** is a collection of distinguishable objects, called its ***members*** or ***elements***. If an object $x$ is a member of a set $S$, we write $x \in S$ (read "$x$ is a member of $S$" or, more briefly, "$x$ is in $S$"). If $x$ is not a member of $S$, we write $x \notin S$. We can describe a set by explicitly listing its members as a list inside braces. For example, we can define a set $S$ to contain precisely the numbers 1, 2, and 3 by writing $S = \{1, 2, 3\}$. Since 2 is a member of the set $S$, we can write $2 \in S$, and since 4 is not a member, we have $4 \notin S$. A set cannot contain the same object more than once,[1] and its elements are not ordered. Two sets $A$ and $B$ are ***equal***, written $A = B$, if they contain the same elements. For example, $\{1, 2, 3, 1\} = \{1, 2, 3\} = \{3, 2, 1\}$.

We adopt special notations for frequently encountered sets:

- $\emptyset$ denotes the ***empty set***, that is, the set containing no members.
- $\mathbb{Z}$ denotes the set of ***integers***, that is, the set $\{\ldots, -2, -1, 0, 1, 2, \ldots\}$.
- $\mathbb{R}$ denotes the set of ***real numbers***.
- $\mathbb{N}$ denotes the set of ***natural numbers***, that is, the set $\{0, 1, 2, \ldots\}$.[2]

---

[1] A variation of a set, which can contain the same object more than once, is called a ***multiset***.

[2] Some authors start the natural numbers with 1 instead of 0. The modern trend seems to be to start with 0.

If all the elements of a set $A$ are contained in a set $B$, that is, if $x \in A$ implies $x \in B$, then we write $A \subseteq B$ and say that $A$ is a ***subset*** of $B$. A set $A$ is a ***proper subset*** of $B$, written $A \subset B$, if $A \subseteq B$ but $A \neq B$. (Some authors use the symbol "$\subset$" to denote the ordinary subset relation, rather than the proper-subset relation.) For any set $A$, we have $A \subseteq A$. For two sets $A$ and $B$, we have $A = B$ if and only if $A \subseteq B$ and $B \subseteq A$. For any three sets $A$, $B$, and $C$, if $A \subseteq B$ and $B \subseteq C$, then $A \subseteq C$. For any set $A$, we have $\emptyset \subseteq A$.

We sometimes define sets in terms of other sets. Given a set $A$, we can define a set $B \subseteq A$ by stating a property that distinguishes the elements of $B$. For example, we can define the set of even integers by $\{x : x \in \mathbb{Z} \text{ and } x/2 \text{ is an integer}\}$. The colon in this notation is read "such that." (Some authors use a vertical bar in place of the colon.)

Given two sets $A$ and $B$, we can also define new sets by applying ***set operations***:

- The ***intersection*** of sets $A$ and $B$ is the set

  $$A \cap B = \{x : x \in A \text{ and } x \in B\} .$$

- The ***union*** of sets $A$ and $B$ is the set

  $$A \cup B = \{x : x \in A \text{ or } x \in B\} .$$

- The ***difference*** between two sets $A$ and $B$ is the set

  $$A - B = \{x : x \in A \text{ and } x \notin B\} .$$

Set operations obey the following laws:

**Empty set laws:**

$$\begin{aligned} A \cap \emptyset &= \emptyset , \\ A \cup \emptyset &= A . \end{aligned}$$

**Idempotency laws:**

$$\begin{aligned} A \cap A &= A , \\ A \cup A &= A . \end{aligned}$$

**Commutative laws:**

$$\begin{aligned} A \cap B &= B \cap A , \\ A \cup B &= B \cup A . \end{aligned}$$

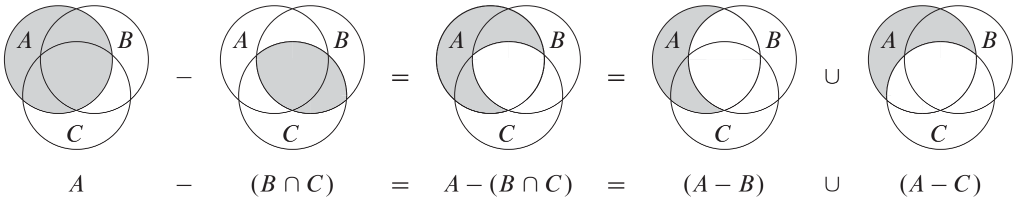


**Figure B.1** A Venn diagram illustrating the first of DeMorgan's laws (B.2). Each of the sets $A$, $B$, and $C$ is represented as a circle.

### Associative laws:

$$
\begin{aligned}
A \cap (B \cap C) &= (A \cap B) \cap C \,, \\
A \cup (B \cup C) &= (A \cup B) \cup C \,.
\end{aligned}
$$

### Distributive laws:

$$
\begin{aligned}
A \cap (B \cup C) &= (A \cap B) \cup (A \cap C) \,, \\
A \cup (B \cap C) &= (A \cup B) \cap (A \cup C) \,.
\end{aligned}
\tag{B.1}
$$

### Absorption laws:

$$
\begin{aligned}
A \cap (A \cup B) &= A \,, \\
A \cup (A \cap B) &= A \,.
\end{aligned}
$$

### DeMorgan's laws:

$$
\begin{aligned}
A - (B \cap C) &= (A - B) \cup (A - C) \,, \\
A - (B \cup C) &= (A - B) \cap (A - C) \,.
\end{aligned}
\tag{B.2}
$$

Figure B.1 illustrates the first of DeMorgan's laws, using a ***Venn diagram***: a graphical picture in which sets are represented as regions of the plane.

Often, all the sets under consideration are subsets of some larger set $U$ called the ***universe***. For example, if we are considering various sets made up only of integers, the set $\mathbb{Z}$ of integers is an appropriate universe. Given a universe $U$, we define the ***complement*** of a set $A$ as $\overline{A} = U - A = \{x : x \in U \text{ and } x \notin A\}$. For any set $A \subseteq U$, we have the following laws:

$$
\begin{aligned}
\overline{\overline{A}} &= A \,, \\
A \cap \overline{A} &= \emptyset \,, \\
A \cup \overline{A} &= U \,.
\end{aligned}
$$

We can rewrite DeMorgan's laws (B.2) with set complements. For any two sets $B, C \subseteq U$, we have

$$\begin{aligned}\overline{B \cap C} &= \overline{B} \cup \overline{C} , \\ \overline{B \cup C} &= \overline{B} \cap \overline{C} .\end{aligned}$$

Two sets $A$ and $B$ are ***disjoint*** if they have no elements in common, that is, if $A \cap B = \emptyset$. A collection $\mathcal{S} = \{S_i\}$ of nonempty sets forms a ***partition*** of a set $S$ if

- the sets are ***pairwise disjoint***, that is, $S_i, S_j \in \mathcal{S}$ and $i \neq j$ imply $S_i \cap S_j = \emptyset$, and
- their union is $S$, that is,

$$S = \bigcup_{S_i \in \mathcal{S}} S_i .$$

In other words, $\mathcal{S}$ forms a partition of $S$ if each element of $S$ appears in exactly one $S_i \in \mathcal{S}$.

The number of elements in a set is the ***cardinality*** (or ***size***) of the set, denoted $|S|$. Two sets have the same cardinality if their elements can be put into a one-to-one correspondence. The cardinality of the empty set is $|\emptyset| = 0$. If the cardinality of a set is a natural number, we say the set is ***finite***; otherwise, it is ***infinite***. An infinite set that can be put into a one-to-one correspondence with the natural numbers $\mathbb{N}$ is ***countably infinite***; otherwise, it is ***uncountable***. For example, the integers $\mathbb{Z}$ are countable, but the reals $\mathbb{R}$ are uncountable.

For any two finite sets $A$ and $B$, we have the identity

$$|A \cup B| = |A| + |B| - |A \cap B| , \tag{B.3}$$

from which we can conclude that

$$|A \cup B| \leq |A| + |B| .$$

If $A$ and $B$ are disjoint, then $|A \cap B| = 0$ and thus $|A \cup B| = |A| + |B|$. If $A \subseteq B$, then $|A| \leq |B|$.

A finite set of $n$ elements is sometimes called an ***n-set***. A 1-set is called a ***singleton***. A subset of $k$ elements of a set is sometimes called a ***k-subset***.

We denote the set of all subsets of a set $S$, including the empty set and $S$ itself, by $2^S$; we call $2^S$ the ***power set*** of $S$. For example, $2^{\{a,b\}} = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}$. The power set of a finite set $S$ has cardinality $2^{|S|}$ (see Exercise B.1-5).

We sometimes care about setlike structures in which the elements are ordered. An ***ordered pair*** of two elements $a$ and $b$ is denoted $(a, b)$ and is defined formally as the set $(a, b) = \{a, \{a, b\}\}$. Thus, the ordered pair $(a, b)$ is *not* the same as the ordered pair $(b, a)$.