

# Appendix A

## Notation and Preliminaries

This appendix sums up important notation, definitions, and key lemmas that are not the main focus of the lecture.

### A.1 Numbers and Sets

In this lecture, zero is not a natural number:  $0 \notin \mathbb{N}$ ; we just write  $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$  whenever we need it.  $\mathbb{Z}$  denotes the integers,  $\mathbb{Q}$  the rational numbers, and  $\mathbb{R}$  the real numbers. We use  $\mathbb{R}^+ = \{x \in \mathbb{R} \mid x > 0\}$  and  $\mathbb{R}_0^+ = \{x \in \mathbb{R} \mid x \geq 0\}$ , with similar notation for  $\mathbb{Z}$  and  $\mathbb{Q}$ .

Rounding down  $x \in \mathbb{R}$  is denoted by  $\lfloor x \rfloor := \max\{z \in \mathbb{Z} \mid z \leq x\}$  and rounding up by  $\lceil x \rceil := \min\{z \in \mathbb{Z} \mid z \geq x\}$ .

For  $n \in \mathbb{N}_0$ , we define  $[n] := \{0, \dots, n-1\}$ , and for a set  $M$  and  $k \in \mathbb{N}_0$ ,  $\binom{M}{k} := \{N \subseteq M \mid |N| = k\}$  is the set of all subsets of  $M$  that contain exactly  $k$  elements.

### A.2 Graphs

A finite set of *vertices*, also referred to as *nodes*  $V$  together with *edges*  $E \subseteq \binom{V}{2}$  defines a *graph*  $G = (V, E)$ . Unless specified otherwise,  $G$  has  $n = |V|$  vertices and  $m = |E|$  edges. Since edges are sets of exactly two vertices  $e = \{v, w\} \subseteq V$ ,<sup>1</sup> our graphs have no *loops*, are *undirected* and have no *parallel edges*. This definition does not include *edge weights*, either. All of this together is equivalent of saying that we deal with *simple* graphs.

If  $e = \{v, w\} \in E$ , the vertices  $v$  and  $w$  are *adjacent*, and  $e$  is *incident* to  $v$  and  $w$ , furthermore,  $e' \in E$  is *adjacent* to  $e$  if  $e \cap e' \neq \emptyset$ . The *neighborhood* of  $v$  is

$$N_v := \{w \in V \mid \{v, w\} \in E\}, \quad (\text{A.1})$$

i.e., the set of vertices adjacent to  $v$ . The *degree* of  $v$  is

$$\delta_v := |N_v|, \quad (\text{A.2})$$

---

<sup>1</sup>Still, we occasionally write edges as tuples:  $e = (v, w)$ .

the size of  $v$ 's neighborhood. We denote by

$$\Delta := \max_{v \in V} \delta_v \quad (\text{A.3})$$

the maximum degree in  $G$ .

A  $v_1$ - $v_d$ -*path*  $p$  is a set of edges  $p = \{\{v_1, v_2\}, \{v_2, v_3\}, \dots, \{v_{d-1}, v_d\}\}$  such that  $|\{e \in p \mid v \in e\}| \leq 2$  for all  $v \in V$ .  $p$  has  $|p|$  *hops*, and we call  $p$  a *cycle* if it visits all of its nodes exactly twice. The *distance* between  $v, w \in V$  is

$$\text{dist}(v, w) := \min\{|p| \mid p \text{ is a } v\text{-}w\text{-path}\}, \quad (\text{A.4})$$

which gives rise to the *diameter*  $D$  of  $G$ ,

$$D := \max_{v, w \in V} \text{dist}(v, w), \quad (\text{A.5})$$

the maximum pairwise distance between nodes.

### A.2.1 Weighted Graphs

A *weighted graph* is a graph  $(V, E)$  together with weighting function  $W: E \rightarrow \mathbb{R}$ ; we write  $G = (V, E, W)$ . An edge  $e \in E$  has *weight*  $W(e)$ , and an edge set  $E' \subseteq E$  has weight  $W(E') := \sum_{e \in E'} W(e)$ . Observe that since paths are sets of edges, this definition captures the weight of a path  $p$ :  $W(p) = \sum_{e \in p} W(e)$ .

In weighted graphs, distances are more complex than in simple graphs, because there are several measures: the smallest weight of a path, and the number of hops. The *distance* between  $v, w \in V$  is the weight of a shortest  $v$ - $w$ -path

$$\text{dist}(v, w) := \min\{W(p) \mid p \text{ is a } v\text{-}w\text{-path}\}, \quad (\text{A.6})$$

and the *hop distance* is the smallest number of hops required to attain a shortest  $v$ - $w$ -path is

$$\text{hop}(v, w) := \min\{|p| \mid p \text{ is } v\text{-}w\text{-path} \wedge W(p) = \text{dist}(v, w)\}. \quad (\text{A.7})$$

Note that shortest paths can be very long in terms of hops, even if there is some (non-shortest) path with few hops: Even if  $\{v, w\} \in E$ ,  $\text{hop}(v, w) = n - 1$  is still possible (think about a circle with one heavy and  $n - 1$  light edges).

### A.2.2 Trees and Forests

A *forest* is a cycle-free graph, and a *tree* is a connected forest. Trees have  $n - 1$  edges and a unique path between any pair of vertices. The tree  $T = (V, E)$  is *rooted* if it has a designated root node  $r \in V$ ; in which case it has a *depth*  $d$

$$d := \max_{v \in V} \text{dist}(r, v), \quad (\text{A.8})$$

which is the maximum distance from any node to the root node.

### A.2.3 Cuts

Given a graph  $G = (V, E)$  and distinct vertices  $s, t \in V$ , an  $s$ - $t$  *cut* is a non-trivial partition of the vertices  $V_s \dot{\cup} V_t = V$ , such that  $s \in V_s$  and  $t \in V_t$ . The *weight* of the cut is  $|E \cap (V_s \times V_t)|$ , i.e., the number of the edges connecting a vertex in  $V_s$  with to a vertex in  $V_t$  (in a weighted graph, the weight of the cut is the sum of those edges' weights). Alternatively, cuts can be represented as only the set  $V_s$  ( $V_t = V \setminus V_s$ ), or as the edge set  $E \cap (V_s \times V_t)$ .

### A.3 Logarithms and Exponentiation

Logarithms are base 2 logarithms, unless specified otherwise. Iterated exponentiation is denoted by  ${}^a b$ , which is the  $a$ -fold ( $a \in \mathbb{N}_0$ ) exponentiation of  $b$ :

$${}^a b := \begin{cases} 1 & \text{if } a = 0 \\ b({}^{a-1}b) & \text{otherwise.} \end{cases} \quad (\text{A.9})$$

This is commonly referred to as *power tower*  ${}^a 2 = 2^{2^{\cdot^{\cdot^2}}}$ .  $\log_b^* x$  answers the inverse question of how often the logarithm has to be iteratively applied to end up with a result of at most 1:

$$\log_b^* x := \begin{cases} 0 & \text{if } x \leq 1 \\ 1 + \log_b^*(\log_b x) & \text{otherwise.} \end{cases} \quad (\text{A.10})$$

A simple inductive check confirms  $\log_b^* {}^a b = a$ .

### A.4 Probability Theory

We use some basic tools from probability theory in order to analyze randomized algorithms. The first of these tools states that the probability that at least one of  $k$  events occur is bounded by the sum of the individual events' probabilities.

**Theorem A.1** (Union Bound). *Let  $\mathcal{E}_i$ ,  $i \in [k]$  be events. Then*

$$P \left[ \bigcup_{i \in [k]} \mathcal{E}_i \right] \leq \sum_{i \in [k]} p_i, \quad (\text{A.11})$$

*which is tight if  $\mathcal{E}_{[k]}$  are disjoint.*

Another key property is that expectation is compatible with summation:

**Theorem A.2** (Linearity of Expectation). *Let  $X_i$  for  $i \in [k]$  denote random variables. Then*

$$\mathbb{E} \left[ \sum_{i \in [k]} X_i \right] = \sum_{i \in [k]} \mathbb{E}[X_i]. \quad (\text{A.12})$$

Markov's inequality and Chernoff's bound both bound the probability that a random variable attains a very different value from its expected value. The preconditions for Markov's inequality are much weaker than those for Chernoff's bound, but the latter is stronger than the former.

**Theorem A.3** (Markov's Inequality). *Let  $X$  be a positive random variable (in fact,  $P[X \geq 0] = 1$  and  $P[X = 0] < 1$  suffice). Then for any  $K > 1$ ,*

$$P[X \geq K\mathbb{E}[X]] \leq \frac{1}{K}. \quad (\text{A.13})$$

**Theorem A.4** (Chernoff's Bound). *Let  $X = \sum_{i \in [k]} X_i$  be the sum of  $k$  independent Bernoulli variables (i.e., 0-1-variables). Then we have, for any  $0 < \delta \leq 1$ ,*

$$P[X \geq (1 + \delta)\mathbb{E}[X]] \leq e^{-\delta^2\mathbb{E}[X]/3}, \text{ and} \quad (\text{A.14})$$

$$P[X \leq (1 - \delta)\mathbb{E}[X]] \leq e^{-\delta^2\mathbb{E}[X]/2}. \quad (\text{A.15})$$

The concept of something happening *with high probability (w.h.p.)*, i.e., with probability at least  $1 - n^{-c}$ , is the following. First, the larger your input  $n$ , the larger the probability that the event occurs. Second,  $c$  can be picked at will, meaning that the probabilities can be picked as close to 1 as desired. This is useful for randomized algorithms. Suppose you have a randomized algorithm  $\mathcal{A}$  that succeeds with probability  $p$ . If you want to use  $\mathcal{A}$  several times (e.g. to construct a new algorithm) the probability that all of these calls succeed decreases. But if *each* call of  $\mathcal{A}$  succeeds w.h.p. and there only are polynomially many of them, you can use the union bound and pick a large enough  $c$  to show that *all* calls of  $\mathcal{A}$  succeed w.h.p. as well.

**Definition A.5** (With high Probability). *The event  $\mathcal{E}$  occurs with high probability (w.h.p.) if  $P[\mathcal{E}] \geq 1 - 1/n^c$  for any fixed choice of  $1 \leq c \in \mathbb{R}$ . Note that  $c$  typically is considered a constant in terms of the  $\mathcal{O}$ -notation.*

## A.5 Asymptotic Notation

We require asymptotic notation to reason about the complexity of algorithms. This section is adapted from Chapter 3 of Cormen et al. [CLR90]. Let  $f, g: \mathbb{N}_0 \rightarrow \mathbb{R}$  be functions.

### A.5.1 Definitions

$\mathcal{O}(g(n))$  is the set containing all functions  $f$  that are bounded from above by  $cg(n)$  for some constant  $c > 0$  and for all sufficiently large  $n$ , i.e.  $f(n)$  is *asymptotically bounded from above* by  $g(n)$ .

$$\mathcal{O}(g(n)) := \{f(n) \mid \exists c \in \mathbb{R}^+, n_0 \in \mathbb{N}_0: \forall n \geq n_0: 0 \leq f(n) \leq cg(n)\} \quad (\text{A.16})$$

The counterpart of  $\mathcal{O}(g(n))$  is  $\Omega(g(n))$ , the set of functions *asymptotically bounded from below* by  $g(n)$ , again up to a positive scalar and for sufficiently large  $n$ :

$$\Omega(g(n)) := \{f(n) \mid \exists c \in \mathbb{R}^+, n_0 \in \mathbb{N}_0: \forall n \geq n_0: 0 \leq cg(n) \leq f(n)\} \quad (\text{A.17})$$

If  $f(n)$  is bounded from below by  $c_1g(n)$  and from above by  $c_2g(n)$  for positive scalars  $c_1$  and  $c_2$  and sufficiently large  $n$ , it belongs to the set  $\Theta(g(n))$ ; in this case  $g(n)$  is an *asymptotically tight* bound for  $f(n)$ . It is easy to check that  $\Theta(g(n))$  is the intersection of  $\mathcal{O}(g(n))$  and  $\Omega(g(n))$ .

$$\Theta(g(n)) := \{f(n) \mid \exists c_1, c_2 \in \mathbb{R}^+, n_0 \in \mathbb{N}_0: \forall n \geq n_0: 0 \leq c_1g(n) \leq f(n) \leq c_2g(n)\} \quad (\text{A.18})$$

$$f(n) \in \Theta(g(n)) \Leftrightarrow f \in (\mathcal{O}(g(n)) \cap \Omega(g(n))) \quad (\text{A.19})$$

For example,  $n \in \mathcal{O}(n^2)$  but  $n \notin \Omega(n^2)$  and thus  $n \notin \Theta(n^2)$ .<sup>2</sup> But  $3n^2 - n + 5 \in \mathcal{O}(n^2)$ ,  $3n^2 - n + 5 \in \Omega(n^2)$ , and thus  $3n^2 - n + 5 \in \Theta(n^2)$  for  $c_1 = 1$ ,  $c_2 = 3$ , and  $n_0 = 4$ .

In order to express that an asymptotic bound is not tight, we require  $o(g(n))$  and  $\omega(g(n))$ .  $f(n) \in o(g(n))$  means that for any positive constant  $c$ ,  $f(n)$  is strictly smaller than  $cg(n)$  for sufficiently large  $n$ .

$$o(g(n)) := \{f(n) \mid \forall c \in \mathbb{R}^+ : \exists n_0 \in \mathbb{N}_0 : \forall n \geq n_0 : 0 \leq f(n) < cg(n)\} \quad (\text{A.20})$$

As an example, consider  $\frac{1}{n}$ . For arbitrary  $c \in \mathbb{R}^+$ ,  $\frac{1}{n} < c$  we have that for all  $n \geq \frac{1}{c} + 1$ , so  $\frac{1}{n} \in o(1)$ . A similar concept exists for lower bounds that are not asymptotically tight;  $f(n) \in \omega(g(n))$  if for any positive scalar  $c$ ,  $cg(n) < f(n)$  as soon as  $n$  is large enough.

$$\omega(g(n)) := \{f(n) \mid \forall c \in \mathbb{R}^+ : \exists n_0 \in \mathbb{N}_0 : \forall n \geq n_0 : 0 \leq cg(n) < f(n)\} \quad (\text{A.21})$$

$$f(n) \in \omega(g(n)) \Leftrightarrow g(n) \in o(f(n)) \quad (\text{A.22})$$

### A.5.2 Properties

We list some useful properties of asymptotic notation, all taken from Chapter 3 of Cormen et al. [CLR90]. The statements in this subsection hold for all  $f, g, h: \mathbb{N}_0 \rightarrow \mathbb{R}$ .

#### Transitivity

$$f(n) \in \mathcal{O}(g(n)) \wedge g(n) \in \mathcal{O}(h(n)) \Rightarrow f(n) \in \mathcal{O}(h(n)), \quad (\text{A.23})$$

$$f(n) \in \Omega(g(n)) \wedge g(n) \in \Omega(h(n)) \Rightarrow f(n) \in \Omega(h(n)), \quad (\text{A.24})$$

$$f(n) \in \Theta(g(n)) \wedge g(n) \in \Theta(h(n)) \Rightarrow f(n) \in \Theta(h(n)), \quad (\text{A.25})$$

$$f(n) \in o(g(n)) \wedge g(n) \in o(h(n)) \Rightarrow f(n) \in o(h(n)), \text{ and} \quad (\text{A.26})$$

$$f(n) \in \omega(g(n)) \wedge g(n) \in \omega(h(n)) \Rightarrow f(n) \in \omega(h(n)). \quad (\text{A.27})$$

#### Reflexivity

$$f(n) \in \mathcal{O}(f(n)), \quad (\text{A.28})$$

$$f(n) \in \Omega(f(n)), \text{ and} \quad (\text{A.29})$$

$$f(n) \in \Theta(f(n)). \quad (\text{A.30})$$

#### Symmetry

$$f(n) \in \Theta(g(n)) \Leftrightarrow g(n) \in \Theta(f(n)). \quad (\text{A.31})$$

#### Transpose Symmetry

$$f(n) \in \mathcal{O}(g(n)) \Leftrightarrow g(n) \in \Omega(f(n)), \text{ and} \quad (\text{A.32})$$

$$f(n) \in o(g(n)) \Leftrightarrow g(n) \in \omega(f(n)). \quad (\text{A.33})$$

<sup>2</sup>We write  $f(n) \in \mathcal{O}(g(n))$  unlike some authors who, by abuse of notation, write  $f(n) = \mathcal{O}(g(n))$ .  $f(n) \in \mathcal{O}(g(n))$  emphasizes that we are dealing with *sets* of functions.

## A.6 Approximation

We require a precise definition of approximation algorithms in order to specify the (im)possibility of such solutions. Let  $\mathcal{A}$  be an algorithm,  $G$  the input,  $f$  the function to be approximated,  $\mathcal{A}(G)$  the cost or value of the valid approximation, and  $\alpha$  the approximation degree.

If for all inputs  $G$ ,

$$f(G) \leq \mathcal{A}(G) \leq \alpha \cdot f(G), \quad (\text{A.34})$$

then we call  $\mathcal{A}$  to be an  $\alpha$ -approximation algorithm.

We will apply this in the context of distances, diameters, arboricities, etc.

## Bibliography

[CLR90] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. The MIT Press, Cambridge, MA, 1990.