A Primer to Randomness

Karl Bringmann

In this course we design several randomized algorithms. Since this requires basic knowledge of randomness and the analysis of random variables, in this document we give an overview of the necessary concepts.

1 Random Variables and Events

We assume familiarity with the notion of a *random variable*. Intuitively, random variables are a formal way to describe random numbers. For a random variable X, we write $\mathbb{P}[X = n]$ for the probability that the outcome of X is n. The sum over all possible outcomes of X is $\sum_n \mathbb{P}[X = n] = 1$. Examples of random variables are as follows.

- *Random bit:* We determine a bit X by a coin toss. A coin can come up as either heads or tails, which we write as 1 or 0. That is, X takes values in $\{0, 1\}$ and we have $\mathbb{P}[X = 0] = \mathbb{P}[X = 1] = \frac{1}{2}$.
- *Throwing 2 coins:* Throw 2 coins and let X be their sum. Then X takes values in $\{0, 1, 2\}$ and $\mathbb{P}[X = 0] = \mathbb{P}[X = 2] = \frac{1}{4}$ and $\mathbb{P}[X = 1] = \frac{1}{2}$.
- *Throwing 10 coins:* Throw 10 coins and count the number X of coins that come up heads. Then X takes values in $\{0, 1, \dots, 10\}$ and $\mathbb{P}[X = n] = \binom{10}{n}2^{-10}$, where $\binom{a}{b}$ denotes a binomial coefficient.
- *Throwing coins until we see heads:* Repeatedly throw a coin until it comes up heads, and let X be the total number of coin tosses. Then X takes values in $\mathbb{N}$ and $\mathbb{P}[X = n] = 2^{-n}$ for any $n \geq 1$.

We also assume familiarity with the notion of an *event*. For an event $\mathcal{E}$ we write $\mathbb{P}[\mathcal{E}]$ for the probability that event $\mathcal{E}$ occurs. To state examples of events, let us fix the random variable X to be the third example from above, so X counts the number of heads among 10 coin tosses.

- $X = 0$: This event has probability $\mathbb{P}[X = 0] = \binom{10}{0} \cdot 2^{-10} = 2^{-10} \approx 0.001$.
- $X \leq 1$: This event has probability
$$\mathbb{P}[X \leq 1] = \mathbb{P}[X = 0] + \mathbb{P}[X = 1] = \left(\binom{10}{0} + \binom{10}{1}\right) \cdot 2^{-10} = 11 \cdot 2^{-10} \approx 0.01.$$
- *X is even:* This event has probability $\mathbb{P}[X \text{ is even}] = \frac{1}{2}$ (exercise).

We say that random variable X is *uniformly distributed* in U if X takes values in U and for all $u \in U$ we have $\mathbb{P}[X = u] = 1/|U|$.

Excursion 1. *For randomized algorithms, we assume access to a method* $\textsc{rand}()$ *that returns a random bit. Often we also assume access to a method* $\textsc{rand}(n)$*, which returns a random number in* $\{1, \dots, n\}$*. Therefore, random variables and events as in the above examples come up naturally in the analysis of randomized algorithms.*

2 Union Bound and Expectation

Lemma 2 (Union Bound)**.** *For any events $\mathcal{E}, \mathcal{E}'$ we have $\mathbb{P}[\mathcal{E} \text{ or } \mathcal{E}'] \le \mathbb{P}[\mathcal{E}] + \mathbb{P}[\mathcal{E}']$.*

For instance, for the example events from the last section we obtain

$$\mathbb{P}[X \text{ is even or } X \le 1] \;\le\; \mathbb{P}[X \text{ is even}] + \mathbb{P}[X \le 1] \;\le\; \frac{1}{2} + 11 \cdot 2^{-10} \approx 0.501.$$

Excursion 3. *In the analysis of randomized algorithms, we typically consider* error events*, in which certain desired properties do not hold. We want to show that the combined probability of all error events is small, in order to show that the algorithm has a large success probability. To this end, it often suffices to bound the probability of each single error event, and then to use the union bound to get an upper bound on the probability that any error event occurs.*

Definition 4. *The* expectation *of a random variable X is*

$$\mathbb{E}[X] := \sum_n n \cdot \mathbb{P}[X = n],$$

where the sum goes over all possible values that X can attain.

The example random variables from the last section have the following expectations.

- *Random bit:* $\mathbb{E}[X] = 0 \cdot \mathbb{P}[X = 0] + 1 \cdot \mathbb{P}[X = 1] = 0 \cdot \frac{1}{2} + 1 \cdot \frac{1}{2} = \frac{1}{2}$.
- *Throwing 2 coins:* $\mathbb{E}[X] = 0 \cdot \frac{1}{4} + 1 \cdot \frac{1}{2} + 2 \cdot \frac{1}{4} = 1$.
- *Throwing 10 coins:* $\mathbb{E}[X] = 5$.
- *Throwing coins until we see heads:* $\mathbb{E}[X] = \sum_{n=1}^{\infty} n \cdot 2^{-n} = 2$.

To analyze expectations of sums of random variables (as in the case of throwing 10 coins), the following is a very useful property.

Lemma 5 (Linearity of Expectation)**.** *For any random variables X, Y we have $\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y]$. More generally, for random variables X, Y and constants $\alpha, \beta \in \mathbb{R}$ we have $\mathbb{E}[\alpha \cdot X + \beta \cdot Y] = \alpha \cdot \mathbb{E}[X] + \beta \cdot \mathbb{E}[Y]$.*

For instance, since the expectation of throwing 1 coin is $\frac{1}{2}$, the expected sum of 10 coins is $\frac{10}{2} = 5$.

3 Concentration Inequalities

3.1 Markov Inequality

We often want to bound the probability that a random variable attains a very large value. Markov's inequality yields a basic bound of this kind.

Lemma 6 (Markov Inequality)**.** *For any random variable $X \ge 0$ and any $t > 0$ we have*

$$\mathbb{P}[X \ge t] \;\le\; \frac{\mathbb{E}[X]}{t}.$$

Proof. This follows from

$$\mathbb{E}[X] \;=\; \sum_{n \ge 0} n \cdot \mathbb{P}[X = n] \;\ge\; \sum_{n \ge t} n \cdot \mathbb{P}[X = n] \;\ge\; \sum_{n \ge t} t \cdot \mathbb{P}[X = n] \;=\; t \cdot \mathbb{P}[X \ge t]. \qquad \square$$

3.2 Chebyshev Inequality

An improved concentration bound can be obtained from the *variance*.

Definition 7. *The* variance *of a random variable X is*

$$\mathrm{Var}[X] \;:=\; \mathbb{E}\big[(X-\mathbb{E}[X])^2\big] \;=\; \mathbb{E}\big[X^2\big]-\mathbb{E}[X]^2.$$

Lemma 8 (Chebyshev Inequality)**.** *For any random variable X and any $t>0$ we have*

$$\mathbb{P}\big[|X-\mathbb{E}[X]|\geq t\big] \;\leq\; \frac{\mathrm{Var}[X]}{t^2}.$$

Proof. Using Markov, we have

$$\mathbb{P}\big[|X-\mathbb{E}[X]|\geq t\big] \;=\; \mathbb{P}\big[(X-\mathbb{E}[X])^2\geq t^2\big] \;\leq\; \frac{\mathbb{E}\big[(X-\mathbb{E}[X])^2\big]}{t^2}.$$

Note that here we use Markov on the random variable $Y:=(X-\mathbb{E}[X])^2$, which indeed satisfies $Y\geq 0$. □

3.3 Chernoff Bound

The strongest concentration inequalities can be obtained for sums of *independent* random variables.

Definition 9. *Two random variables X,Y are* independent *if for any values x,y we have*

$$\mathbb{P}[X=x \text{ and } Y=y] \;=\; \mathbb{P}[X=x]\cdot\mathbb{P}[Y=y].$$

More generally, random variables $X_1,\dots,X_n$ are independent if for any $x_1,\dots,x_n$ we have

$$\mathbb{P}\big[X_1=x_1 \text{ and } \dots \text{ and } X_n=x_n\big] \;=\; \mathbb{P}[X_1=x_1]\cdot\ldots\cdot\mathbb{P}[X_n=x_n].$$

Let X_1,X_2 be two coin flips, that is, we flip a coin to determine $X_1\in\{0,1\}$ and then we flip a coin again to determine $X_2\in\{0,1\}$. Naturally, the random variables X_1 and X_2 are independent. For an example of non-independence, let $X:=X_1+X_2$. Then the random variables X_1 and X are not independent, since

$$\mathbb{P}[X=0 \text{ and } X_1=1] \;=\; 0 \;\neq\; \frac{1}{4}\cdot\frac{1}{2} \;=\; \mathbb{P}[X=0]\cdot\mathbb{P}[X_1=1].$$

For a non-trivial example of independence, let $Y:=X_1\oplus X_2$ (the XOR operation applied to X_1 and X_2). Then one can check that the random variables X_1 and Y are independent.

Independent random variables generally behave nicely, for instance expectations and products commute, as shown by the following lemma whose proof is left as an exercise.

Lemma 10. *Independent random variables $X_1,\dots,X_n$ satisfy*

$$\mathbb{E}[X_1\cdot\ldots\cdot X_n] \;=\; \mathbb{E}[X_1]\cdot\ldots\cdot\mathbb{E}[X_n].$$

Lemma 11 (Chernoff Bound)**.** *Let $X_1,\dots,X_n$ be independent random variables taking values in $\{0,1\}$, and let $X:=\sum_{j=1}^n X_j$. Then for all $t>0$ we have*

$$\mathbb{P}\big[X\geq\mathbb{E}[X]+t\big] \;\leq\; \exp\left(-\frac{2t^2}{n}\right).$$

By symmetry, the same holds for $\mathbb{P}[X\leq\mathbb{E}[X]-t]$.

The Chernoff bound is a very strong concentration inequality; we omit the proof. Many variations with different error bounds as well as generalizations that somewhat relax the independence assumption are known.

3.4 Additional Remarks

- Often we use Chebyshev's inequality together with the simple inequality $\mathrm{Var}[X] \leq \mathbb{E}[X^2]$, so we bound

$$\mathbb{P}\big[|X - \mathbb{E}[X]| \geq t\big] \;\leq\; \frac{\mathbb{E}[X^2]}{t^2}.$$

- Following the proof of Chebyshev's inequality, but replacing the exponent 2 by any even integer $p \geq 2$, yields the following more general inequality. For any random variable X and numbers $t > 0$ and any even integer $p \geq 2$ we have

$$\mathbb{P}\big[|X - \mathbb{E}[X]| \geq t\big] \;\leq\; \frac{\mathbb{E}\big[(X - \mathbb{E}[X])^p\big]}{t^p}.$$

  This is a concentration inequality in terms of higher moments of X.

- An *indicator random variable* (or short indicator variable) is a random variable I taking values in $\{0,1\}$. Note that we have $\mathbb{E}[I] = 0 \cdot \mathbb{P}[I = 0] + 1 \cdot \mathbb{P}[I = 1] = \mathbb{P}[I = 1]$. Also note that an indicator variable I is essentially the same as the event $I = 1$ (one could say that I indicates the event $I = 1$). In fact, indicator variables and events are two notations for the same objects. Indicator variables naturally appear in the formulation of the Chernoff bound; compare with Lemma 11.

- So far we implicitly assumed that X is discrete, more precisely we typically consider random variables whose values are integers. Rarely in this course we will also encounter *continuous* random variables, for instance a random real number $X \in [0,1]$. The definitions and facts in this document still hold for this random variable, after replacing sums by integrals.

4 Boosting

We now discuss how to use concentration inequalities in the context of algorithm design. Consider the problem of computing a function f, that is, for a given instance I we want to compute the number $f(I)$. Suppose that we have already designed a randomized algorithm $\mathcal{A}$ that for any instance I computes a number $\mathcal{A}(I)$ such that $\mathbb{E}[\mathcal{A}(I)] = f(I)$. Such an algorithm is called an *unbiased estimator* for f. Unbiased estimators are often easy to design, but they do not correspond to our usual understanding of approximation. We would much rather like to have a guaranteee of the form $\mathcal{A}(I) \in \big[(1-\varepsilon)f(I), (1+\varepsilon)f(I)\big]$ for some $\varepsilon \in (0,1)$, that is, a *multiplicative approximation* of f.

In this section, we analyze the probability that algorithm $\mathcal{A}$ computes a multiplicative approximation, and we show how to *boost* this probability by repeated calls to algorithm $\mathcal{A}$.

4.1 Boosting via Chebyshev

Note that Chebyshev's inequality implies a statement about multiplicative approximations, as it shows that

$$\mathbb{P}\big[|\mathcal{A}(I) - f(I)| \geq \varepsilon \cdot f(I)\big] \;\leq\; \frac{\mathrm{Var}[\mathcal{A}(I)]}{\varepsilon^2 f(I)^2}. \tag{1}$$

This shows that algorithm $\mathcal{A}$ computes a multiplicative approximation with error probability bounded by $p_{\mathcal{A}} := \frac{\mathrm{Var}[\mathcal{A}(I)]}{\varepsilon^2 f(I)^2}$. However, not all unbiased estimators work: It can very well be that the error bound $p_{\mathcal{A}}$ is larger than 1, which makes guarantee (1) useless!

A remedy comes from the method of boosting, where we average over m runs of our algorithm. We remark that the outcomes of different calls to our random source RAND() are assumed to be independent. In particular, running the same algorithm twice (with fresh randomness) yields independent outcomes.

So denote by $\mathcal{A}_1(I), \ldots, \mathcal{A}_m(I)$ independent runs of algorithm $\mathcal{A}$ on instance I. Consider the algorithm $\mathcal{B}$ which returns the value

$$\mathcal{B}(I) \;:=\; \frac{1}{m} \cdot \big(\mathcal{A}_1(I) + \ldots + \mathcal{A}_m(I)\big).$$

In the following we analyze this algorithm. Linearity of expectation states that for (not necessarily independent!) random variables X, Y and constants α, β we have $\mathbb{E}[\alpha X + \beta Y] = \alpha\mathbb{E}[X] + \beta\mathbb{E}[Y]$. Thus,

$$\mathbb{E}[\mathcal{B}(I)] \;=\; \frac{1}{m} \cdot \big(\mathbb{E}[\mathcal{A}_1(I)] + \ldots + \mathbb{E}[\mathcal{A}_m(I)]\big) \;=\; f(I),$$

so also algorithm $\mathcal{B}$ is an unbiased estimator for f.

For the variance $\mathrm{Var}[\alpha X + \beta Y]$ in general there is no such clean bound. However, for any random variable X and any constant α we have $\mathrm{Var}[\alpha X] = \alpha^2 \mathrm{Var}[X]$. Moreover, if X, Y are *independent*, then for any constants α, β we have $\mathrm{Var}[\alpha X + \beta Y] = \alpha^2\mathrm{Var}[X] + \beta^2\mathrm{Var}[Y]$. This yields

$$\mathrm{Var}[\mathcal{B}(I)] \;=\; \frac{1}{m^2} \cdot \big(\mathrm{Var}[\mathcal{A}_1(I)] + \ldots + \mathrm{Var}[\mathcal{A}_m(I)]\big) \;=\; \frac{1}{m} \cdot \mathrm{Var}[\mathcal{A}(I)].$$

Plugging these bounds into Chebyshev's inequality yields

$$\mathbb{P}\big[|\mathcal{B}(I) - f(I)| \ge \varepsilon \cdot f(I)\big] \;\le\; \frac{\mathrm{Var}[\mathcal{B}(I)]}{\varepsilon^2 f(I)^2} \;=\; \frac{p_{\mathcal{A}}}{m}.$$

Thus, the error bound $p_{\mathcal{B}}$ of algorithm $\mathcal{B}$ is smaller by a factor m than the error bound $p_{\mathcal{A}}$ of algorithm $\mathcal{A}$, and this comes from averaging m runs of algorithm $\mathcal{A}$. In other words, we boosted the error probability by a factor m.

We remark that this method is even applicable if $p_{\mathcal{A}} \ge 1$, where a single call to algorithm $\mathcal{A}$ does not give any approximation guarantee. By averaging over $\lceil 10 \cdot p_{\mathcal{A}} \rceil$ runs we decrease the error bound to $p_{\mathcal{B}} = 0.1$. This is an improvement from no approximation guarantee at all to a multiplicative approximation with good probability! Such a situation is the main use of boosting via Chebyshev's inequality.

4.2 Boosting via Chernoff

Consider the same situation as before, but additionally assume that algorithm $\mathcal{A}$ already has a reasonable error probability, specifically an error probability of at most 1/3, that is,

$$\mathbb{P}\big[|\mathcal{A}(I) - f(I)| \ge \varepsilon \cdot f(I)\big] \;\le\; \frac{1}{3}.$$

Our goal is to design an algorithm $\mathcal{C}$ that has error probability δ, for some desired $\delta > 0$. Boosting via Chebyshev tells us that averaging over $\mathcal{O}(1/\delta)$ calls to algorithm $\mathcal{A}$ yields error probability δ. We will now see that $\mathcal{O}(\log(1/\delta))$ calls suffice.[1]

Let $\mathcal{A}_1(I), \ldots, \mathcal{A}_m(I)$ be independent runs of algorithm $\mathcal{A}$ on instance I. Algorithm $\mathcal{C}(I)$ simply returns the median of the numbers $\mathcal{A}_1(I), \ldots, \mathcal{A}_m(I)$ (that is, it sorts these numbers and picks the middle element).

We analyze algorithm $\mathcal{C}$ in the following claim.

Claim 12. *We have*

$$\mathbb{P}\big[|\mathcal{C}(I) - f(I)| \ge \varepsilon \cdot f(I)\big] \;\le\; \exp\Big(-\frac{m}{20}\Big).$$

This claim implies that with $m = \lceil 20 \log(1/\delta) \rceil$ repetitions we obtain error probability δ. It remains to prove the claim.

Proof. We define random variables $X_1, \ldots, X_m \in \{0, 1\}$ such that $X_i = 1$ if any only if $|\mathcal{A}_i(I) - f(I)| \le \varepsilon \cdot f(I)$. That is, X_i indicates whether the i-th call succeeds. By assumption, we have $\mathbb{E}[X_i] = \mathbb{P}[X_i = 1] \ge \frac{2}{3}$. Hence, for $X := \sum_{i=1}^m X_i$ we have by linearity of expectation

$$\mathbb{E}[X] \;=\; \mathbb{E}[X_1] + \ldots + \mathbb{E}[X_m] \;\ge\; \frac{2m}{3}.$$

Observe that algorithm $\mathcal{C}$ succeeds if the majority of the calls to algorithm $\mathcal{A}$ succeed. It follows that

$$\mathbb{P}\big[|\mathcal{C}(I) - f(I)| \ge \varepsilon \cdot f(I)\big] \;\le\; \mathbb{P}\Big[X \le \frac{m}{2}\Big] \;\le\; \mathbb{P}\Big[X \le \mathbb{E}[X] - \frac{m}{6}\Big] \;\le\; \exp\Big(-\frac{2m}{36}\Big) \;\le\; \exp\Big(-\frac{m}{20}\Big),$$

where we used the Chernoff bound with $t := \frac{m}{6}$. □

[1]In asymptotic notation when we write $\log x$ we typically mean $\log(2 + x)$, to avoid issues for very small x. Moreover, all logarithms are base 2.

4.3 Combined Boosting

We often use a combination of boosting via Chebyshev and boosting via Chernoff: First, we use boosting via Chebyshev to reduce the error probability to $\frac{1}{3}$, and then we use boosting via Chernoff to reduce the error probability further to any desired δ. This yields the following.

Lemma 13 (Combined Boosting)**.** *Let $\varepsilon, \delta \in (0,1)$. Suppose that algorithm $\mathcal{A}$ is an unbiased estimator of f (that is, $\mathbb{E}[\mathcal{A}(I)] = f(I)$ for any instance I). Let $p_{\mathcal{A}} = p_{\mathcal{A}}(I) := \frac{\mathrm{Var}[\mathcal{A}(I)]}{\varepsilon^2 f(I)^2}$ for any instance I.*

Then there is an algorithm $\mathcal{B}$ that combines the results of $\mathcal{O}((p_{\mathcal{A}}+1)\log(1/\delta))$ calls to algorithm $\mathcal{A}$ and has error probability δ, that is, for any instance I we have

$$\mathbb{P}\big[\big|\mathcal{B}(I) - f(I)\big| \geq \varepsilon \cdot f(I)\big] \leq \delta.$$

Proof. We consider an intermediate algorithm $\mathcal{A}'$ that averages over $\lceil 3p_{\mathcal{A}} \rceil$ runs of algorithm $\mathcal{A}$. From boosting via Chebyshev we see that algorithm $\mathcal{A}'$ has error probability $1/3$. Therefore, we can use boosting via Chernoff on algorithm $\mathcal{A}'$ to further improve the error probability to δ, by computing the median of $\lceil 20\log(1/\delta) \rceil$ calls to algorithm $\mathcal{A}'$. In total, we combine

$$\lceil 3p_{\mathcal{A}} \rceil \cdot \lceil 20\log(1/\delta) \rceil = \mathcal{O}((p_{\mathcal{A}}+1)\log(1/\delta))$$

runs of algorithm $\mathcal{A}$. □

In particular, if $\mathrm{Var}[\mathcal{A}(I)] = \mathcal{O}(f(I)^2)$, then we obtain a multiplicative approximation algorithm $\mathcal{B}$ with

$$\mathbb{P}\big[\big|\mathcal{B}(I) - f(I)\big| \geq \varepsilon \cdot f(I)\big] \leq \delta$$

that uses $\mathcal{O}(\frac{1}{\varepsilon^2}\log(1/\delta))$ calls to algorithm $\mathcal{A}$.

5 Hashing

This material will be needed in Lecture 2.

By $[n]$ we denote the set $\{1, 2, \ldots, n\}$. Consider a function

$$h\colon [n] \to [m].$$

This maps an item from the universe $[n]$ into one of m buckets, specifically item i is mapped to bucket $h(i)$. In our situation the function h will be picked uniformly at random from a family of hash functions $\mathcal{H}$. Thus, every value $h(i)$ is a random variable. We will usually ensure that the random variable $h(i)$ is uniformly distributed in $[m]$.

Note that we do not assume that the random variables $h(1), \ldots, h(n)$ are independent. Therefore, there are two extremes of hash functions:

- *Constant function:* We set $h(1) = h(2) = \ldots = h(n)$ to be the same random value, or in other words $\mathcal{H}$ is the family of all constant functions. Note that each value $h(i)$ is uniformly distributed, but their values are highly dependent. Such a function h can be stored in small space, using only $\lceil \log m \rceil$ bits. However, since all items are mapped to the same bucket, it fails to capture the intuitive idea of a hash function randomly distributing items into buckets.

- *Truly random function:* We let $h(1), h(2), \ldots, h(n)$ be independent random variables that are uniformly distributed in $[m]$, in other words $\mathcal{H}$ is the family of *all* functions from $[n]$ to $[m]$. This yields a great hash function from the perspective of using randomness to distribute items into buckets. However, in order to store h we need to store n numbers, or $n \log m$ bits, which is prohibitive in the context of sublinear algorithms.

We will next see a concept that captures the best of the above two extremes.

5.1 Pairwise Independent Hashing

Definition 14. *Random variables $X_1, \dots, X_n$ are* pairwise independent *if for any $i \neq i'$ the random variables X_i and $X_{i'}$ are independent. In other words, $X_1, \dots, X_n$ are pairwise independent if for any j, j' and any distinct $i, i' \in [n]$ we have*

$$\mathbb{P}[X_i = j \text{ and } X_{i'} = j'] = \mathbb{P}[X_i = j] \cdot \mathbb{P}[X_{i'} = j'].$$

Comparing with Definition 9, we see that for $n = 2$, pairwise independence and independence coincide, while for $n > 2$, pairwise independence is a weaker property than independence.

Definition 15. *A family $\mathcal{H}$ of functions mapping $[n]$ to $[m]$ is a* family of pairwise independent hash functions *if for uniformly random $h \in \mathcal{H}$ it holds that (1) the random variables $h(1), \dots, h(n)$ are pairwise independent and (2) for any fixed i the hash value $h(i)$ is uniformly distributed in $[m]$.*

Note that truly random hash functions (where all $h(i)$ are independent) form a family of pairwise independent hash functions. A very useful construction of pairwise independent hash functions is given next.

Lemma 16. *Let $m = p$ be a prime with $m \geq n$, and let $\mathcal{H}_2$ be the set of all functions of the form*

$$h(i) = (a \cdot i + b) \bmod p,$$

for any $a, b \in [p]$. Then a function $h \in \mathcal{H}_2$ can be represented by the pair $(a, b) \in [p]^2$, using $2\lceil \log p \rceil = 2\lceil \log m \rceil$ bits. The family $\mathcal{H}_2$ is a family of pairwise independent hash functions.

Proof. We make use of the fact that the numbers modulo p form a field $\mathbb{F}_p$ and thus have algebraic structure. Note that $\mathcal{H}_2$ is the family of lines in $\mathbb{F}_p$. Given (i, j) and (i', j') with $i \neq i'$, there exists exactly one line through the points (i, j) and (i', j'). Hence, for random $h \in \mathcal{H}_2$ the probability $\mathbb{P}[h(i) = j \text{ and } h(i') = j']$ is equal to $1/|\mathcal{H}_2| = 1/p^2 = 1/m^2$. Moreover, for fixed i, after picking a, the random choice of b yields a uniformly distributed value $h(i) \in [m]$. □

We remark that it is known that any interval $[n, 2n]$ contains a prime number, and thus the assumption that $m = p$ is a prime is essentially without loss of generality, up to changing m by a factor of at most 2.

Note that the family $\mathcal{H}_2$ combines the best of the two extremes, as it can be stored in small space while satisfying a variant of independence. Moreover, we can sample a hash function $h \in \mathcal{H}_2$ in constant time, using two calls to RAND(p).

Recall that in boosting with Chebyshev (Section 4.1) we use that for independent random variables $X_1, \dots, X_n$ we have $\mathrm{Var}[X_1 + \ldots + X_n] = \mathrm{Var}[X_1] + \ldots + \mathrm{Var}[X_n]$. We next show that this identity already holds for pairwise independent random variables (and thus boosting via Chebyshev also works for pairwise independent calls).

Lemma 17. *For pairwise independent random variables $X_1, \dots, X_n$ we have*

$$\mathrm{Var}[X_1 + \ldots + X_n] \; = \; \mathrm{Var}[X_1] + \ldots + \mathrm{Var}[X_n].$$

Proof. By definition of the variance and linearity of expectation we have

$$\mathrm{Var}[X_1 + \ldots + X_n] \; = \; \mathbb{E}\big[(X_1 + \ldots + X_n)^2\big] - \mathbb{E}[X_1 + \ldots + X_n]^2 \; = \; \sum_{i,j} \mathbb{E}[X_i \cdot X_j] - \sum_{i,j} \mathbb{E}[X_i] \cdot \mathbb{E}[X_j].$$

Since any X_i and X_j for $i \neq j$ are independent, by Lemma 10 we have $\mathbb{E}[X_i \cdot X_j] = \mathbb{E}[X_i] \cdot \mathbb{E}[X_j]$. Hence, all summands with $i \neq j$ cancel. We thus obtain

$$\mathrm{Var}[X_1 + \ldots + X_n] \; = \; \sum_i \big(\mathbb{E}\big[X_i^2\big] - \mathbb{E}[X_i]^2\big) \; = \; \mathrm{Var}[X_1] + \ldots + \mathrm{Var}[X_n].$$ □

5.2 k-Wise Independent Hashing

Generalizing the previous section gives rise to the following definitions and construction.

Definition 18. *Random variables $X_1, \dots, X_n$ are k-*wise independent *if for any distinct $i_1, \dots, i_k$ the random variables $X_{i_1}, \dots, X_{i_k}$ are independent. In other words, $X_1, \dots, X_n$ are k-wise independent if for any $j_1, \dots, j_k$ and any distinct $i_1, \dots, i_k \in [n]$ we have*

$$\mathbb{P}\big[X_{i_1} = j_1 \text{ and } \dots \text{ and } X_{i_k} = j_k\big] = \mathbb{P}[X_{i_1} = j_1] \cdot \ldots \cdot \mathbb{P}[X_{i_k} = j_k].$$

Definition 19. *A family $\mathcal{H}$ of functions mapping $[n]$ to $[m]$ is a* family of k-wise independent hash functions *if for uniformly random $h \in \mathcal{H}$ it holds that (1) the random variables $h(1), \dots, h(n)$ are k-wise independent and (2) for any fixed i the hash value $h(i)$ is uniformly distributed in $[m]$.*

Lemma 20. *Let $m = p$ be a prime with $m \geq n$, and let $\mathcal{H}_k$ be the set of all polynomials of degree at most $k-1$ over $\mathbb{F}_p$. Then a function $h \in \mathcal{H}_k$ can be represented by a tuple $(a_0, \dots, a_{k-1}) \in [p]^k$, using $k\lceil \log p \rceil = k\lceil \log m \rceil$ bits. The family $\mathcal{H}_k$ is a family of k-wise independent hash functions.*

The proof of the above lemma uses that for any k given points there is exactly one polynomial of degree $k-1$ that goes through all these points.