Chapter 2

Linearity of Expectation

Linearity of expectation basically says that the expected value of a sum of random variables is equal to the sum of the individual expectations. Its importance can hardly be overestimated for the area of randomized algorithms and probabilistic methods. Its main power lies in the facts that it

(i) is applicable for sums of *any* random variables (independent or not), and

(ii) that it often allows simple "local" arguments instead of "global" ones.

2.1 Basics

For some given (discrete) probability space Ω any mapping $X : \Omega \to \mathbb{Z}$ is called a (numerical) *random variable*. The *expected value* of X is given by

$$\mathbb{E}[X] = \sum_{\omega \in \Omega} X(\omega) \Pr[\omega] = \sum_{x \in \mathbb{Z}} x \Pr[X = x]$$

provided that $\sum_{x \in \mathbb{Z}} |x| \Pr[X = x]$ converges.

Example 2.1. Let X_1 and X_2 denote two independent rolls of a fair dice. What is the expected value of the sum $X = X_1 + X_2$? We use the definition, calculate and obtain

$$\mathbb{E}[X] = 2 \cdot \frac{1}{36} + 3 \cdot \frac{1}{36} + \cdots + 12 \cdot \frac{1}{36} = 7.$$

As stated already, linearity of expectation allows us to compute the expected value of a sum of random variables by computing the sum of the individual expectations.

Theorem 2.2. *Let $X_1, \ldots, X_n$ be any finite collection of discrete random variables and let $X = \sum_{i=1}^{n} X_i$. Then we have*

$$\mathbb{E}[X] = \mathbb{E}\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \mathbb{E}[X_i].$$

Proof. We use the definition, reorder the sum by its finiteness, and obtain

$$\begin{aligned}
\mathbb{E}[X] &= \sum_{\omega\in\Omega} X(\omega)\Pr[\omega] \\
&= \sum_{\omega\in\Omega} (X_1(\omega) + \cdots + X_n(\omega))\Pr[\omega] \\
&= \sum_{i=1}^{n} \sum_{\omega\in\Omega} X_i(\omega)\Pr[\omega] \\
&= \sum_{i=1}^{n} \mathbb{E}[X_i],
\end{aligned}$$

which was claimed. □

It can be shown that linearity of expectation also holds for countably infinite summations in certain cases. For example, it holds that

$$\mathbb{E}\left[\sum_{i=1}^{\infty} X_i\right] = \sum_{i=1}^{\infty} \mathbb{E}[X_i]$$

if $\sum_{i=1}^{\infty} \mathbb{E}[|X_i|]$ converges.

Example 2.3. Recalling Example 2.1, we first compute $\mathbb{E}[X_1] = \mathbb{E}[X_2] = 1 \cdot 1/6 + \cdots + 6 \cdot 1/6 = 7/2$ and hence

$$\mathbb{E}[X] = \mathbb{E}[X_1] + \mathbb{E}[X_2] = \frac{7}{2} + \frac{7}{2} = 7.$$

Admittedly, in such a trivial example, the power of linearity of expectation can hardly be seen. This should, however, change in the applications to come.

2.2 Applications

2.2.1 Balls Into Bins

Many problems in computer science and combinatorics can be formulated in terms of a Balls into Bins process. We give some examples here. Suppose we have m balls, labeled $i = 1, \ldots, m$ and n bins, labeled $j = 1, \ldots, n$. Each ball is thrown into one of the bin independently and uniformly at random.

Theorem 2.4. *Let X_j denote the number of balls in bin j. Then, for $j = 1, \ldots, m$ we have*

$$\mathbb{E}[X_j] = \frac{m}{n}$$

Proof. Define an indicator variable

$$X_{i,j} = \begin{cases} 1 & \text{ball } i \text{ falls into bin } j, \\ 0 & \text{otherwise,} \end{cases}$$

for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. We have $\Pr[X_{i,j} = 1] = \mathbb{E}[X_{i,j}] = 1/n$ and by linearity of expectation

$$\mathbb{E}[X_j] = \mathbb{E}\left[\sum_{i=1}^{m} X_{i,j}\right] = \sum_{i=1}^{m} \mathbb{E}[X_{i,j}] = \sum_{i=1}^{m} \frac{1}{n} = \frac{m}{n}$$

as claimed. □

Theorem 2.5. *Let X denote the number of empty bins when m balls are thrown independently into n bins. Then we have*

$$\mathbb{E}[X] = n \cdot \left(1 - \frac{1}{n}\right)^m \simeq n \cdot e^{-m/n}.$$

Proof. Define an indicator variable

$$X_{i,j} = \begin{cases} 1 & \text{ball } i \text{ misses bin } j, \\ 0 & \text{otherwise,} \end{cases}$$

for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. We obviously have

$$\Pr[X_{i,j} = 1] = \mathbb{E}[X_{i,j}] = 1 - \frac{1}{n}.$$

For any $j = 1, \ldots, m$, $X_j = \Pi_{i=1}^{m} X_{i,j}$ indicates if all balls have missed bin j. By independence,

$$\mathbb{E}[X_j] = \mathbb{E}[\Pi_{i=1}^{m} X_{i,j}] = \Pi_{i=1}^{m} \mathbb{E}[X_{i,j}] = \left(1 - \frac{1}{n}\right)^m,$$

where we have used $\mathbb{E}[X \cdot Y] = \mathbb{E}[X] \cdot \mathbb{E}[Y]$ for independent X and Y. Finally, linearity of expectation yields

$$\mathbb{E}[X] = \mathbb{E}\left[\sum_{j=1}^{n} X_j\right] = \sum_{j=1}^{n} \mathbb{E}[X_j] = \sum_{j=1}^{n} \left(1 - \frac{1}{n}\right)^m = n \cdot \left(1 - \frac{1}{n}\right)^m \simeq n \cdot e^{-m/n},$$

as claimed. □

Suppose we have $m = n$. By Theorem 2.4 we expect that each bin contains one ball. But by Theorem 2.5 we have $\mathbb{E}[X] = n \cdot (1 - 1/n)^n \simeq n/e$ (for n sufficiently large). This means we expect that a constant fraction of the bins remains empty. At first sight this may seem contradictive, but it is not, because there is also a constant fraction of the bins that contain more than one ball. (Excercise.)

2.2.2 Coupon Collector

The Coupon Collector problem is the following: Suppose there are n types of coupons in a lottery and each lot contains one coupon (with probability $1/n$ each). How many lots have to be bought (in expectation) until we have at least one coupon of each type. (In terms of a Balls into Bins process, the question is, how many balls have to be thrown (in expectation) until no bin is empty.)

Example 2.6. A famous German brand of Haselnusstafeln hides a picture of some football-player in each sweet. How many sweets have to be bought until one has a picture of each player (under the unrealistic assumption that all players are equiprobable)?

Theorem 2.7. *Let X be the number of lots bought until at least one coupon of each type is drawn. Then we have*

$$\mathbb{E}[X] = n \cdot H_n,$$

where $H_n = \sum_{i=1}^{n} 1/i$ denotes the n-th Harmonic number.

Proof. We partition the process of drawing lots into phases. In phase P_i for $i = 1, \ldots, n$ we have already collected $i-1$ distinct coupons and the phase ends once we have drawn i distinct coupons. Let X_i be the number of lots bought in phase P_i.

Suppose we are in phase P_i, then the probability that the next lot terminates this phase is $(n-i+1)/n$. This is because there are $n-(i-1)$ many coupon-types we have not yet collected. Any of those coupons will be the i-th distinct type to be collected (since we have exactly $i-1$ at the moment). These events happen with probability $1/n$, each. These considerations imply that the random variable X_i has geometric distribution with success-probability $(n-i+1)/n$, i.e., $X_i \sim \text{Geo}((n-i+1)/n)$. Its expected value is the reciprocal, i.e., $\mathbb{E}[X_i] = n/(n-i+1)$.

Now we invoke linearity of expectation and obtain

$$\mathbb{E}[X] = \mathbb{E}\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \mathbb{E}[X_i] = \sum_{i=1}^{n} \frac{n}{n-i+1} = n \cdot H_n$$

as claimed. □

Recall that $\log n \leq H_n \leq \log n + 1$. Thus we basically have to buy $n \log n$ lots.

2.2.3 Quicksort

The problem of Sorting is the following: We are given a sequence $x = (x_1, x_2, \ldots, x_n)$ of (pairwise distinct) numbers and are asked to find a permutation π of $(1, 2, \ldots, n)$ such that the sequence $(x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)})$ satisfies $x_{\pi(1)} \leq x_{\pi(2)} \leq \cdots \leq x_{\pi(n)}$. The assumption that the numbers are pairwise distinct can easily be removed, but we consider it for clarity of exposition. Any algorithm for this problem is allowed to ask queries of the type "$a < b$?", called a comparison. Let $\text{rt}_A(x)$ be the number of comparisons of an algorithm A given a sequence x.

The idea of the algorithm Quicksort is to choose some element p from x, called the *pivot*, and divide x into two subsequences x' and x''. The sequence x' contains the elements $x_i < p$ and x'' those $x_i > p$. Quicksort is then called recursively until the input-sequence is empty. The sequence $(\text{Quicksort}(x'), p, \text{Quicksort}(x''))$ is finally returned.

The exposition of Quicksort here is actually not yet a well-defined algorithm, because we have not yet said how we want to choose the pivot element p. This choice drastically affects the running time as we will see shortly. In the sequel let X denote the number of comparisons "$<$" executed by Quicksort.

Deterministic Algorithm

Suppose we choose always the first element in the input-sequence, i.e., $p = x_1$. It is well-known that this variant of Quicksort has the weakness that it may require $\Omega(n^2)$ comparisons.

Observation 2.8. *There is an instance with $X = n(n-1)/2$.*

Algorithm 2.1 QUICKSORT

Input. Sequence $(x_1, x_2, \dots, x_n)$

Output. Sequence $(x_{\pi(1)}, x_{\pi(2)}, \dots, x_{\pi(n)})$

(1) If $n = 0$ return.

(2) Otherwise choose $p \in x$ arbitrarily and remove p from x. Let x' and x'' be two empty sequences.

(3) For $i = 1, \dots, n$, if $x_i < p$ append x_i to x', otherwise append x_i to x''.

(4) Return $(\textsc{Quicksort}(x'), p, \textsc{Quicksort}(x''))$

Proof. Consider $x = (1, 2, \dots, n)$. Then, in step (3), x' remains empty while x'' contains $n-1$ elements. This step requires $n-1$ comparisons. By induction, the recursive calls $\textsc{Quicksort}(x')$ and $\textsc{Quicksort}(x'')$ require 0, respectively $(n-1)(n-2)/2$ comparisons "$<$". Thus, the whole algorithm needs $X = n - 1 + (n-1)(n-2)/2 = n(n-1)/2$ comparisons. □

Randomized Algorithm

Now suppose that we always choose the pivot element equiprobably among the available elements. This gives obviously rise to a Las Vegas algorithm, because we will never compute a wrong result. These choices merely affect the (expected) running time.

Theorem 2.9. *We have that* $\mathbb{E}[X] = 2(n+1)H_n - 4n$.

Proof. Without loss of generality (by renaming the numbers in x), we assume that the original sequence x is a permutation of $(1, 2, \dots, n)$. So, for any $i < j \in \{1, 2, \dots, n\}$ let the random variable $X_{i,j}$ be equal to one if i and j are compared during the course of the algorithm and zero otherwise. The total number of comparisons is hence $X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{i,j}$. Thus, by linearity of expectation,

$$\mathbb{E}[X] = \mathbb{E}\left[\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{i,j}\right] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathbb{E}[X_{i,j}] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \Pr[X_{i,j} = 1],$$

which shows that we have to derive the probability that i and j are compared.

First observe that *each* element will be pivot element in the course of the algorithm *exactly* once. Thus the input x and the random choices of the algorithm induce a random sequence $P = (P_1, P_2, \dots, P_n)$ of pivots.

Fix i and j arbitrarily. When will these elements be compared? We claim that it will be the case if and only if either i or j is the *first* pivot from the set $\{i, \dots, j\}$ in the sequence P. If i and j are compared, then either one must be the pivot and they must be in the same subsequence of x. Thus all previous pivots (if any) must be smaller than i or larger than j, since i and j would end up in different subsequences of x, otherwise. Hence, either i or j is the first pivot in the set $\{i, \dots, j\}$ appearing in P. The converse direction is trivial: If one of i or j is the first pivot from the set $\{i, \dots, j\}$ in P, then i and j are still in the same subsequence of x and will hence be compared.

What is the probability that, say, i is the first pivot of $\{i, \dots, j\}$? Consider the subsequence S of P induced by the elements $\{i, \dots, j\}$. Hence i is the first pivot from the set $\{i, \dots, j\}$ if and only if i is the first element from that set in P, i.e., the first element in S. Since the pivot is uniformly distributed the probability that i is the first element of S is exactly $1/(j-i+1)$. Analogous reasoning for j yields the overall probability $\Pr[X_{i,j} = 1] = 2/(j-i+1)$.

This allows us to continue our calculation

$$\begin{aligned}
\mathbb{E}[X] &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \Pr[X_{i,j} = 1] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} = \sum_{i=1}^{n-1} \sum_{k=2}^{n-i+1} \frac{2}{k} \\
&= \sum_{k=2}^{n} \sum_{i=1}^{n+1-k} \frac{2}{k} = \sum_{k=2}^{n} (n+1-k)\frac{2}{k} = (n+1)\sum_{k=2}^{n} \frac{2}{k} - 2(n-1) \\
&= 2(n+1)H_n - 4n
\end{aligned}$$

and the proof is complete. □

So we have shown that the expected number of comparisons of QUICKSORT is $O(n \log n)$. In the next chapter, we will even strengthen the statement: The number of comparisons is $O(n \log n)$ *with high probability*, i.e., with probability that tends to one as n tends to infinity.