Random Variables and Expectation (II)

Josep Díaz    Maria J. Serna    Conrado Martínez
U. Politècnica de Catalunya

RA-MIRI 2022–2023
Most of the material included here is based on Chapter 13 of Kleinberg & Tardos *Algorithm Design* book.
Waiting for a first success

- A coin is heads with probability $p$ and tails with probability $1 - p$.
- How many independent flips we expect to get heads for the first time?
- Let $X$ the random variable that gives the number of flips until (and including) the first head.

Observe that

$$P[X = j] = (1 - p)^{j-1}p$$

and

$$E[X] = \sum_{j=1}^{\infty} j P[X = j] = \sum_{j=1}^{\infty} (1 - p)^{j-1}p = \frac{p}{1 - p} \sum_{j=1}^{\infty} j(1 - p)^j$$

as $\sum_{j=1}^{\infty} jx^j = \frac{x}{(1-x)^2}$, we have

$$E[X] = \frac{p}{1 - p} \frac{1 - p}{p^2} = \frac{1}{p}$$
A Bernoulli process denotes a sequence of experiments, each of them with binary output: success (1) with probability $p$, and failure (0) with prob. $q = 1 - p$.

A nice thing about Bernoulli distributions: it is natural to define an indicator r.v.

$$X = \begin{cases} 1 & \text{if the output is 1}, \\ 0 & \text{otherwise}. \end{cases}$$

Clearly, $\mathbb{E}[X] = \mathbb{P}[X = 1] = p$
The binomial distribution

A r.v. $X$ has a **Binomial distribution** with parameters $n$ and $p$ ($X \sim \text{Bin}(n, p)$) if $X$ counts the number of successes during $n$ trials, each trial an independent Bernoulli experiment having probability of success $p$.

\[
P[X = k] = \binom{n}{k} p^k (1 - p)^{n-k}.
\]

Let $X \sim \text{Bin}(n, p)$. To compute $\mathbb{E}[X]$, we define indicator r.v. \{\(X_i\)\}_{i=1}^n$, where $X_i = 1$ iff the $i$-th output is 1, otherwise $X_i = 0$, that is, each $X_i$ is the indicator r.v. of a Bernoulli experiment.

Then $X = \sum_{i=1}^n X_i \Rightarrow \mathbb{E}[X] = \mathbb{E}[\sum_{i=1}^n X_i] = \sum_{i=1}^n \mathbb{E}[X_i] = np.$
The Geometric distribution

A r.v. $X$ has a **Geometric distribution** with parameter $p$ ($X \sim \text{Geom}(p)$) if $X$ counts the number of Bernoulli trials until the first success.

If $X \sim \text{Geom}(p)$ then

$\mathbb{P}[X = k] = (1 - p)^{k-1}p$, \\
$\mathbb{E}[X] = \frac{1}{p}$. 
Random generators

Consider a sequential random generator of \( n \) bits, so that the probability that a bit is 1 is \( p \).

- If \( X = \# \) number of 1's in the generated \( n \) bit number, \( X \sim \text{Bin}(n, p) \).
- If \( Y = \# \) bits in the generated number until the first 1, \( Y \sim \text{Geom}(p) \).
Coupon collector

Each box of cereal contains a coupon. There are \( n \) different types of coupons. Assuming all boxes are equally likely to contain each coupon, how many boxes before you have at least 1 coupon of each type?

**Claim**

The expected number of steps is \( \Theta(n \log n) \).

**Proof**

- Phase \( j \) = number of steps between \( j \) and \( j + 1 \) distinct coupons.
- Let \( X_j \) = number of steps you spend in phase \( j \).
- Let \( X = \text{total number of steps, of course,} \)
  \[ X = X_0 + X_1 + \cdots + X_{n-1}. \]
X_j = number of steps you spend in phase j.

- We can consider a Bernoulli experiment that succeeds when we hit one of the still not collected coupons.
- Conditioned on the event that we have already collected j distinct coupons, the probability of success is \( p_j = \frac{n-j}{n} \).
- \( X_j \) counts the time until the Bernoulli process reaches a success, therefore \( X_j \sim \text{Geom}(p_j) \), hence

\[
E[X_j] = \frac{n}{n-j}
\]
Coupon collector

Proof (cont’d)

$X = \text{total number of steps}$

Using linearity of expectations, we have

$$E[X] = E[X_0] + E[X_1] + \cdots + E[X_{n-1}]$$

$$= \sum_{j=0}^{n-1} \frac{n}{n-j} = n \sum_{j=1}^{n} \frac{1}{j} = nH_n = n \ln n + \Theta(n).$$
A randomized approximation algorithm for MAX 3-SAT

A 3-SAT formula is a Boolean formula in CNF such that each clause has exactly 3 literals and each literal corresponds to a different variable.

\((x_2 \lor \overline{x_3} \lor \overline{x_4}) \land (x_2 \lor x_3 \lor \overline{x_4}) \land (\overline{x_1} \lor x_2 \lor x_4) \land (\overline{x_1} \lor \overline{x_2} \lor x_3) \land (x_1 \lor x_2 \lor x_4)\)

MAXIMUM 3-SAT. Given a 3-SAT formula, find a truth assignment that satisfies as many clauses as possible.

The problem is NP-hard. We can try to design a randomized algorithm that produces a good assignment, even if it is not optimal.
A randomized approximation algorithm for MAX 3-SAT

Algorithm. For each variable, flip a fair coin, and set the variable to **true** (1) if it is heads, to **false** (0) otherwise.

Note that a variable gets 1 with probability $\frac{1}{2}$, and this assignment is made independently of the other variables.

What is the expected number of satisfied clauses?

Assume that the 3-SAT formula has $n$ variables and $m$ clauses.

- Let $Z =$ number of clauses satisfied by the random assignment
- For $1 \leq j \leq m$, define the random variables $Z_j = 1$ if clause $j$ is satisfied, 0 otherwise.
- By definition, $Z = \sum_{j=1}^{m} Z_j$.
- $\mathbb{P}[Z_j = 1] = 1 - (1/2)^3 = 7/8$, so $\mathbb{E}[Z_j] = 7/8$. Therefore,

$$\mathbb{E}[Z] = \sum_{j=1}^{m} \mathbb{E}[Z_j] = \frac{7}{8}m$$
A randomized approximation algorithm for MAX 3-SAT

How good is the solution computed by the random algorithm?

- For a 3-CNF formula let $\text{opt}(F)$ be the maximum number of clauses than can be satisfied by an assignment.
- As for any assignment $x$ the number of satisfied clauses is always $\leq \text{opt}(F)$, we have that $\mathbb{E}[Z] \leq \text{opt}(F)$.
- Of course $\text{opt}(F) \leq m$, that is $\frac{7}{8}\text{opt}(F) \leq \frac{7}{8}m = \mathbb{E}[Z]$, then

$$\frac{\text{opt}(F)}{\mathbb{E}[Z]} \leq \frac{8}{7}$$

We have a $\frac{8}{7}$-approximation algorithm for MAX 3-SAT.
The probabilistic method

Claim
For any instance of 3-SAT, there exists a truth assignment that satisfies at least a 7/8 fraction of all clauses.

Proof
For any random variable $X$ there must exist one event $\omega$ for which the measured value $X(\omega)$ is at least as large as the expectation of $X$.

Probabilistic method. [Paul Erdős] Prove the existence of a non-obvious property by showing that a random construction produces it with positive probability.
Random Quicksort

**Input:** An array \( A \) holding \( n \) keys. For simplicity we assumed that all keys are different.

**Output:** \( A \) sorted in increasing order.

I’m assuming that all of you known:

- The Quicksort algorithm which has \( \Theta(n^2) \) cost
- and \( \Theta(n \log n) \) average cost.
- One randomized version randomly sorts the input and then applies the deterministic algorithm, having average running time \( \Theta(n \log n) \)
- Here we consider another randomized version of Quicksort.
procedure RAND-QUICKSORT(A)
    if A.size() <= 3 then
        Sort A using insertion sort
        return A
    end if
    Choose an element $a \in A$ uniformly at random
    Put in $A^-$ all elements $< a$ and in $A^+$ all elements $> a$
    RAND-QUICKSORT($A^-$)
    RAND-QUICKSORT($A^+$)
    $A := A^- \cdot a \cdot A^+$
end procedure

The main difference is that we perform a random partition in each call around the random pivot $a$. 
Example

A = \{1,3,5,6,8,10,12,14,15,16,17,18,20,22,23\}

Ran−Partition of input
The expected running time $T(n)$ of Rand-Quicksort is dominated by the number of comparisons.

Every Rand-Partition has cost

$\Theta(1) + \Theta(\text{number of comparisons})$

If we can count the number of comparisons, we can bound the the total time of Quicksort.

Let $X$ be the number of comparisons made in all calls of Ran-Quicksort

$X$ is a r.v. as it depends of the random choices of the element used to do a Ran-Partition
Note: In the first application of Ran-Partition the selected $a$ compares with all $n - 1$ elements.

Key observation: Any two keys are compared iff one of them is selected as pivot, and they are compared at most one time.
Denote the $i$-th smallest element in the array by $z_i$ and define the indicator r.v.:

$$X_{ij} = \begin{cases} 
1 & \text{if } z_i \text{ is compared to } z_j, \\
0 & \text{otherwise.}
\end{cases}$$

Then, $X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{i,j}$

(this is true because we never compare a pair more than once)

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

$$\mathbb{E}[X_{i,j}] = \mathbb{P}[X_{i,j} = 1] = \mathbb{P}[z_i \text{ is compared to } z_j]$$
If the pivot we choose is between $z_i$ and $z_j$ then we never compare them to each other.

If the pivot we choose is either $z_i$ or $z_j$ then we do compare them.

If the pivot is less than $z_i$ or greater than $z_j$ then both $z_i$ and $z_j$ end up in the same partition and we have to pick another pivot.

So, we can think of this like a dart game: we throw a dart at random into the array: if we hit $z_i$ or $z_j$ then $X_{ij}$ becomes 1, if we hit between $z_i$ and $z_j$ then $X_{ij}$ becomes 0, and otherwise we throw another dart.

At each step, the probability that $X_{ij} = 1$ conditioned on the event that the game ends in that step is exactly $2/(j - i + 1)$. Therefore, overall, the probability that $X_{ij} = 1$ is $2/(j - i + 1)$.
\[ \mathbb{E}[X] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathbb{E}[X_{i,j}] \]

\[ = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} \]

\[ = 2 \cdot \sum_{i=1}^{n} \left( \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n-i+1} \right) \]

\[ < 2 \cdot \sum_{i=1}^{n} \left( \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} \right) \]

\[ = 2 \cdot \sum_{i=1}^{n} H_n = 2 \cdot n \cdot H_n = \Theta(n \log n). \]

Therefore, \( \mathbb{E}[X] \leq 2n \ln n + \Theta(n). \)
Main theorem

**Theorem**

The expected complexity of Ran-Quicksort is $\mathbb{E}[T_n] = \Theta(n \lg n)$. 
Selection and order statistics

**Problem**: Given a list $A$ of $n$ of unordered distinct keys, and an integer $i \in \mathbb{Z}$, $1 \leq i \leq n$, select the element $x \in A$ that is larger than exactly $i - 1$ other elements in $A$.

Notice if:

1. $i = 1 \Rightarrow$ MINIMUM element
2. $i = n \Rightarrow$ MAXIMUM element
3. $i = \left\lfloor \frac{n+1}{2} \right\rfloor \Rightarrow$ the MEDIAN
4. $i = \left\lfloor 0.9 \cdot n \right\rfloor \Rightarrow$ order statistics

Sort $A$ ($\Theta(n \log n)$) and search for $A[i]$ ($\Theta(n)$).

Can we do it in linear time?

Yes, there are deterministic linear time algorithms for selection—but with a bad constant factor.
Quickselect

Given unordered $A[1, \ldots, n]$ return the $i$-th. element

- Quickselect $(A[p, \ldots, q], i)$
- $r = \text{Ran-Partition} (p, q)$ to find position of pivot and partition the array
- if $i = r$ return $A[r]$
- if $i < r$ Quickselect $(A[p, \ldots, r-1], i)$
- else Quickselect $(A[r+1, \ldots, q], i - r)$

Search for $i=2$ in $A$

A

<table>
<thead>
<tr>
<th>m</th>
<th>u</th>
<th>h</th>
<th>e</th>
<th>c</th>
<th>b</th>
<th>k</th>
<th>v</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>

$3 = \text{Ran-Partition}(1, 8)$

<table>
<thead>
<tr>
<th>e</th>
<th>c</th>
<th>b</th>
<th>h</th>
<th>u</th>
<th>v</th>
<th>k</th>
<th>m</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>
Analysis of Quickselect

In the worst-case, the cost of QUICKSELECT is $\Theta(n^2)$. But on average its cost is $\Theta(n)$.

**Theorem**

*Given $A[1, \ldots, n]$ and $i$, the expected number of steps for Quickselect to find the $i$-th element in $A$ is $\mathcal{O}(n)$.*
Analysis of Quickselect

- The algorithm is in phase $j$ when the size of the set under consideration is at most $n(3/4)^j$ but greater than $n(3/4)^{j-1}$.
- We bound the expected number of iterations spent in phase $j$.
- An element is central if at least a quarter of the elements are smaller and at least a quarter of the elements are larger.
- If a central element is chosen as pivot, at least a quarter of the elements are dropped. So, the set shrinks by a $3/4$ factor or better.
- Since half of the elements are central, the probability of choosing as pivot a central element is $1/2$.
- So the expected number of iterations in phase $j$ is 2.
Analysis of Quickselect

- Let $X =$ number of steps taken by the algorithm.
- Let $X_j =$ number of steps in phase $j$. We have $X = X_0 + X_1 + X_2 + \ldots$
- An iteration in phase $j$ requires at most $cn(3/4)^j$ steps, for some constant $c$.
- Therefore, $\mathbb{E}[X_j] \leq 2cn(3/4)^j$ and by linearity of expectation.

$$
\mathbb{E}[X] = \sum_j \mathbb{E}[X_j] \leq \sum_j 2cn \left(\frac{3}{4}\right)^j = 2cn \sum_j \left(\frac{3}{4}\right)^j \leq 8cn
$$
Analysis of Quickselect

We have proved that its average cost is $\Theta(n)$. The proportionality constant depends on the ratio $i/n$. $C_n^{(i)}$, the expected number of comparisons to find the smallest $i$-th element among $n$ is

$$C_n^{(i)} \sim f(\alpha) \cdot n + o(n), \quad \alpha = i/n,$$

$$f(\alpha) = 2 - 2 \left( \alpha \ln \alpha + (1 - \alpha) \ln(1 - \alpha) \right)$$

More precisely, Knuth (1971) proved that

$$C_n^{(i)} = 2 \left( (n + 1)H_n - (n + 3 - j)H_{n+1-j} \right) - (j + 2)H_j + n + 3$$

The maximum average cost corresponds to finding the median ($i = \lfloor n/2 \rfloor$); then we have

$$C_n^{(\lfloor n/2 \rfloor)} = 2(\ln 2 + 1)n + o(n).$$
CMT considers divide-and-conquer recurrences of the following type:

\[ F_n = t_n + \sum_{0 \leq j < n} \omega_{n,j} F_j, \quad n \geq n_0 \]

for some positive integer \( n_0 \), a function \( t_n \), called the \textit{toll function}, and a sequence of \textit{weights} \( \omega_{n,j} \geq 0 \). The weights must satisfy two conditions:

1. \( W_n = \sum_{0 \leq j < n} \omega_{n,j} \geq 1 \) (at least one recursive call).
2. \( Z_n = \sum_{0 \leq j < n} \frac{j}{n} \cdot \frac{\omega_{n,j}}{W_n} < 1 \) (the size of the subinstances is a fraction of the size of the original instance).

The next step is to find a \textit{shape function} \( \omega(z) \), a continuous function approximating the discrete weights \( \omega_{n,j} \).
The Continuous Master Theorem

Definition

Given the sequence of weights \( \omega_{n,j} \), \( \omega(z) \) is a shape function for that set of weights if

1. \( \int_0^1 \omega(z) \, dz \geq 1 \)
2. there exists a constant \( \rho > 0 \) such that

\[
\sum_{0 \leq j < n} \left| \omega_{n,j} - \int_{j/n}^{(j+1)/n} \omega(z) \, dz \right| = O(n^{-\rho})
\]

A simple trick that works very often, to obtain a convenient shape function is to substitute \( j \) by \( z \cdot n \) in \( \omega_{n,j} \), multiply by \( n \) and take the limit for \( n \to \infty \).

\[
\omega(z) = \lim_{n \to \infty} n \cdot \omega_{n,z \cdot n}
\]
The Continuous Master Theorem

The extension of discrete functions to functions in the real domain is immediate, e.g., $j^2 \rightarrow z^2$. For binomial numbers one might use the approximation

$$\binom{z \cdot n}{k} \sim \frac{(z \cdot n)^k}{k!}.$$ 

The continuation of factorials to the real numbers is given by Euler’s Gamma function $\Gamma(z)$ and that of harmonic numbers by $\Psi$ function: $\Psi(z) = \frac{d \ln \Gamma(z)}{dz}$.

For instance, in quicksort’s recurrence all weights are equal: $\omega_{n,j} = \frac{2}{n}$. Hence a simple valid shape function is $\omega(z) = \lim_{n \rightarrow \infty} n \cdot \omega_{n,z,n} = 2$. 
The Continuous Master Theorem

**Theorem (Roura, 1997)**

Let $F_n$ satisfy the recurrence

$$F_n = t_n + \sum_{0 \leq j < n} \omega_{n,j} F_j,$$

with $t_n = \Theta(n^a(\log n)^b)$, for some constants $a \geq 0$ and $b > -1$, and let $\omega(z)$ be a shape function for the weights $\omega_{n,j}$. Let $\mathcal{H} = 1 - \int_0^1 \omega(z)z^a \, dz$ and $\mathcal{H}' = -(b + 1) \int_0^1 \omega(z)z^a \ln z \, dz$. Then

$$F_n = \begin{cases} \frac{t_n}{\mathcal{H}} + o(t_n) & \text{if } \mathcal{H} > 0, \\ \frac{t_n}{\mathcal{H}'} \ln n + o(t_n \log n) & \text{if } \mathcal{H} = 0 \text{ and } \mathcal{H}' \neq 0, \\ \Theta(n^\alpha) & \text{if } \mathcal{H} < 0, \end{cases}$$

where $\alpha = \alpha$ is the unique non-negative solution of the equation

$$1 - \int_0^1 \omega(z)z^\alpha \, dz = 0.$$
Solving Quicksort’s Recurrence

We apply CMT to quicksort’s recurrence with the set of weights \(\omega_{n,j} = 2/n\) and toll function \(t_n = n - 1\). As we have already seen, we can take \(\omega(z) = 2\), and the CMT applies with \(a = 1\) and \(b = 0\). All necessary conditions to apply CMT are met. Then we compute

\[
H = 1 - \int_0^1 2z \, dz = 1 - z^2 \bigg|_{z=0}^{z=1} = 0,
\]

hence we will have to apply CMT’s second case and compute

\[
H' = - \int_0^1 2z \ln z \, dz = \frac{z^2}{2} - z^2 \ln z \bigg|_{z=0}^{z=1} = \frac{1}{2}.
\]

Finally,

\[
q_n = \frac{n \ln n}{1/2} + o(n \log n) = 2n \ln n + o(n \log n)
\]

\[
= 1.386 \ldots n \log_2 n + o(n \log n).
\]
Analyzing Quickselect

Let us now consider the analysis of the expected cost $C_n$ of Quickselect when sought rank $i$ takes any value between 1 and $n$ with identical probability. Then

$$C_n = n + o(1) + \frac{1}{n} \sum_{1 \leq k \leq n} \mathbb{E}[\text{remaining number of comp.} \mid \text{pivot is the } k\text{-th element}] ,$$

as the pivot will be the $k$-th smallest element with probability $1/n$ for all $k$, $1 \leq k \leq n$. 
Analyzing Quickselect

The probability that \( i = k \) is \( 1/n \), then no more comparisons are need since we would be done. The probability that \( i < k \) is \((k - 1)/n\), then we will have to make \( C_{k-1} \) comparisons. Similarly, with probability \((n - k)/n\) we have \( i > k \) and we will then make \( C_{n-k} \) comparisons. Thus

\[
C_n = n + \Theta(1) + \frac{1}{n} \sum_{1 \leq k \leq n} \frac{k-1}{n} C_{k-1} + \frac{n-k}{n} C_{n-k}
\]

\[
= n + \Theta(1) + \frac{2}{n} \sum_{0 \leq k < n} \frac{k}{n} C_k.
\]

Applying the CMT with the shape function

\[
\lim_{n \to \infty} n \cdot \frac{2}{n} \frac{z \cdot n}{\frac{z \cdot n}{n}} = 2z
\]

we obtain \( H = 1 - \int_0^1 2z^2 \, dz = 1/3 > 0 \) and \( C_n = 3n + o(n) \).