Set 03: Average Case Analysis and Randomized Algorithms

CS240: Data Structures and Data Management

Jérémy Barbay

Outline

Random Variables and Probability Distributions

Given a random source R,

- a **random variable** X is
- the **probability distribution** of a variable X is

Properties of Probability Distributions

Consider two random variables X and Y on a finite domain $D = \{x_1, \ldots, x_n\}$, and a probability distribution $(p_i)_{i \in D}$:

- $\forall i \in D, p_i \in$
- $\sum_{i \in D} p_i =$
- $\Pr[X = i \text{ and } Y = j] =$
- $\Pr[X = i \text{ or } Y = j] =$
- $\Pr[X = i \text{ or } X = j] =$

Operations on Random Variables

- Combination of two random variables is a random variable.
- The probability distribution is not always trivial though.

Example

- Roll two dices, sum the values X and Y of their top faces.
- The sum $X + Y$ has values in

i	2	3	4	5	6	7
$\Pr[X + Y = i]$						
i	12	11	10	9	8	7

Expectation

The **Expectation** of a random variable X is

$$E[X] = \sum_x x \cdot \Pr(X = x).$$

Example

The expected value of a roll of a fair dice is 3.5:

$$\begin{aligned} E[X] &= \sum_{i=1}^{6} i \cdot \Pr(\text{dice rolls to side } i) \\ &= \frac{1+2+3+4+5+6}{6} \\ &= \frac{21}{6} = \frac{7}{2} = 3.5 \end{aligned}$$

Expectation is Linear

The expectation is linear:

$$E[X + Y] = E[X] + E[Y]$$

Example

The expected value of the sum of two dice rolls is

Summary

- A probability is a real number in $[0, 1]$
- The distribution of the combination of two random variables can be counter-intuitive.
- Expectation is linear: $E[X + Y] = E[X] + E[Y]$

Outline

Input Distributions

In many cases, the worst case complexity is irrelevant, and statistical information about the instances is available:

Example

- The average lenght k of a query is 3.2 words.
  $\Rightarrow$ don't optimize algorithms for large k.
- On average, each keyword matches $E(n) = 10,800,000$ pages.
  $\Rightarrow$ algorithms must be optimized for large values of n.

Output Distribution

Given

- a deterministic algorithm,
- and a random distribution $\mathcal{D}$ on the input,

one can compute its average performance.

Given

- a random distribution
- several algorithms,

one can compare the average performances using the same techniques and notations as for the worst case analysis.

Randomized Algorithms

What's a randomized algorithm?
Two definitions:

A deterministic algorithm is a particular case of a randomized algorithm.

Running time of a Randomized Algorithm

- The **running time** $T_A(x, R)$ **of a randomized algorithm** A for a particular input x and a random source R is
- The **expected running time** $T_A^{(exp)}(x)$ **of a randomized algorithm** A for a particular input x and a random source R is .
- The **worst-case expected running time** $T_A^{(exp)}(n)$ **of a randomized algorithm** is

Advantages of Randomized Algorithm

- In security, cryptography.
- to "smoothen" real world behaiour and amortize costs.
- stronger model.

But...

Summary

- Often, average case more interesting than worst case.
- Randomized algorithm = distribution on det. algorithms.
- Randomized is stronger than Deterministic.

Outline

General QuickSort

Recall the QuickSort algorithm from CS134:

```
QuickSort(from,to):
  if to>from
  | i:=Partition(from,to);
  | QuickSort(from,i-1);
  | QuickSort(i+1,to);
```

Specific QuickSort

The following implementation of the Partition chooses deterministically the right-most element as the pivot.

```
Partition(from,to):
  pivot:=A[to];
  i:=from-1;
  for j:=from to to
  | if A[j]<=pivot
  | | i:=i+1
  | | swap(A[j],A[i])
  return i;
```

Running Time

- **Worst-case:**
- **Best-case:**
- **Average-case:**

Randomizing Partition:

Pick the pivot randomly:

```
RandomizedPartition(from,to):
  swap(A[to],A[random(from,to)]);
  pivot:=A[to];
  i:=from-1;
  for j:=from to to
  | if A[j]<=pivot
  | | i:=i+1
  | | swap(A[j],A[i])
  return i;
```

Randomized Analysis

- If the values are all equal, the algorithm runs in time
- Suppose that input numbers $z_1 < z_2 < \cdots < z_n$ (given in some other order) are distinct.

Theorem

Randomized QuickSort performs on average $\boxed{}$ *comparisons when the values are all distinct.*

Proof: Expected number of comparisons

Let's note

- X = total number of comparisons
- $X_{i,j}$ = number of comparisons between z_i and z_j.

Then:

$$\begin{aligned} X_{i,j} &\in \{0,1\} \\ E(X_{i,j}) &= 0 \times \Pr[X_{i,j}=0] + 1 \times \Pr[X_{i,j}=1] \\ &= \Pr[X_{i,j}=1] \\ X &= \sum_{i<j} X_{i,j} \\ E(X) &= \sum_{i<j} E(X_{i,j}) \end{aligned}$$

Proof: Expected number of comparisons (cont')

Consider $X_{i,j}$, suppose $i < j$:

- While the pivot $z_k \notin [z_i, z_j]$,
- If a pivot $z_k \in]z_i, z_j[$ is chosen,
- If z_i or z_j is chosen as pivot,

$$\begin{aligned} E(X_{i,j}) &= \\ &= \end{aligned}$$

Proof: Expected number of comparisons (cont'')

$$\begin{aligned} E[X] &= \sum_{i<j} E[X_{i,j}] = \sum_{i<j} \frac{2}{(j-i+1)} \\ &= 2\sum_{i=1}^{n} \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{i} \\ &\le 2\sum_{i=1}^{n} \int_{x=1}^{i} \frac{1}{x} dx = 2\sum_{i=1}^{n} [\ln x]_{x=1}^{i} \\ &= 2\sum_{i=1}^{n} \ln i \in O(n \log n) \end{aligned}$$

The expected number of comparisons of QuickSort is

$\square$.

Summary

- Det. QuickSort performs ops in worst case.
- Det. QuickSort performs ops on average on the uniform distribution on permutations.
- Rand. QuickSort performs ops on average on its randomness.