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 a function which associates to each event of R a value for X.

Example

- The roll of a dice with six faces is a random event.
- The value on the up face is a random variable, on the domain $\{1, 2, 3, 4, 5, 6\}$.

- the **probability distribution** of a variable X is the function associating a probability to each possible value of X.

Example

If the dice is fair, the probability distribution of X in function of $i \in \{1, 2, 3, 4, 5, 6\}$ is $\Pr(X = i) = 1/6$.

Properties of Probability Distributions

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

- $\forall i \in D, p_i \in [0, 1]$
- $\sum_{i \in D} p_i = 1$
- $\Pr[X = i \text{ and } Y = j] = p_i \times p_j$
- $\Pr[X = i \text{ or } Y = j] = p_i + p_j$
- $\Pr[X = i \text{ or } X = j] = p_i + p_j - p_i p_j$

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 $\{2, \ldots, 12\}$.

i	2	3	4	5	6	7
$\Pr[X + Y = i]$	$1/36$	$2/36$	$3/36$	$4/36$	$5/36$	$6/36$
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:

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

$$E[\sum_i X_i] = \sum_i E[X_i]$$

Example

The expected value of the sum of two dice rolls is
$E[X + Y] = E[X] + E[Y] = 2 \cdot 3.5 = 7$

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:

- Trafic Jams
- Internet Routing
- Compression Schemes

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:

- Algorithm using Randomized Instructions.
- Probabilistic Distribution on Deterministic Algorithms.

A deterministic algorithm is a particular case of a randomized algorithm.
The reverse is not true: randomized algorithms are more powerfull than deterministic ones.

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 a random variable $T_A(x, R)$.
- 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 expectation of $T_A(x, R)$.
- The **worst-case expected running time** $T_A^{(exp)}(n)$ **of a randomized algorithm** is a function of the size n of the input:

$$T_A^{(\mathrm{exp})}(n) = \max\{\, T_A^{(exp)}(x) \mid |x| = n\}$$

Advantages of Randomized Algorithm

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

But...Finding good Random sources is difficult!

Summary

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

Outline

Basics on probabilities

- Random Variables and Probability Distributions
- Expectation

Average Case Analysis

- Input Distributions
- Randomized Algorithms

Deterministic and Randomized QuickSort

- General QuickSort
- Randomized QuickSort
- Randomized Analysis

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:**
  - when all the elements of the array are the same: $\Theta(n^2)$
  - when the array is already sorted: $\Theta(n^2)$
- **Best-case:**
  - if we always find the median value.
  - $\Theta(n \log n)$
- **Average-case:** (Over all permutations of n elements)
  - $\Theta(n \log n)$

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 $O(n^2)$.
- Suppose that input numbers $z_1 < z_2 < \cdots < z_n$ (given in some other order) are distinct.

Theorem

Randomized QuickSort performs on average $\boxed{O(n \log n)}$ *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 $z_k \notin [z_i, z_j]$,
  $X_{i,j}$ stays unchanged.
- If a pivot $z_k \in]z_i, z_j[$ is chosen,
  z_i and z_j are separated and will never be compared
  (Hence $X_{i,j} = 0$)
- If z_i or z_j is chosen as pivot,
  then and only then, $X_{i,j} = 1$

$$
\begin{aligned}
E(X_{i,j}) &= \Pr[X_{i,j} = 1] \\
&= \frac{2}{j - i + 1}
\end{aligned}
$$

Proof: Expected number of comparisons (cont'')

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

The expected number of comparisons of QuickSort is $O(n \log n)$ $\square$.

Summary

- Det. QuickSort performs $O(n^2)$ ops in worst case.
- Det. QuickSort performs $O(n \log n)$ ops on average on the uniform distribution on permutations.
- Rand. QuickSort performs $O(n \log n)$ ops on average on its randomness.