W4231: Analysis of Algorithms

9/21/1999 (revised 9/23)

- Probability
- Randomized Select

Probability

In making computations about probability, we always refer to

- A sample space S.
- A probability distribution $\mathbf{Pr}[\cdot]$.

S is a finite set, containing all the "elementary events" that can happen

$\mathbf{Pr}$ maps S into non-negative reals, and $\sum_{a \in S} \mathbf{Pr}[a] = 1$.

$\mathbf{Pr}[a]$ is the probability of a.

Example

We want to model what happens when we flip a coin three times.

The sample space is the set of eight elementary events $S = \{HHH, HHT, \ldots, TTT\}$.

The probability of each elementary event is $1/8$.

When all the elements of S have the same probability, $\mathbf{Pr}$ is called the *uniform distribution*.

Events

An event is a subset $A \subseteq S$.

The probability of A is defined as $\mathbf{Pr}[A] = \sum_{a \in A} \mathbf{Pr}[a]$.

Example: what is the probability of having two heads when flipping a coin three times.

Define A as the set of elementary events with 2 heads $\{HHT, HTH, THH\}$.

Compute $\mathbf{Pr}[A] = \mathbf{Pr}[HHT] + \mathbf{Pr}[HTH] + \mathbf{Pr}[THH] = 1/8 + 1/8 + 1/8 = 3/8$.

Why Random Files Cannot Be Compressed

Fix a compression algorithm CA.

Generate a random file F of length n bits.

Claim: The probability that $CA(F)$ is of length $\geq n - k$ is at least $1 - 2^{-k}$.

For example, there is a 99.6% chance that a random file be shortened by 8 bits or less.

Proof

The sample space is the set of files of length n. There are 2^n such files.

We assume files are uniformly distributed. Each file has probability 2^{-n} of being generated.

There are $\leq 2^l$ files F such that the length of $CA(F)$ is l.

There are $\leq 2^{n-k} - 1$ files F such that the length of $CA(F)$ is $< n - k$.

The probability that one such file is generated is $< 2^{-k}$.

Random Variable

A random variable X is a way of associating a real number to each element of the sample space S.

Formally, a random variable is a function mapping S into the real numbers.

For a random variable X and a value v, the event $(X = v)$ is the set of elements $a \in S$ such that $X(a) = v$.

Example

Consider again S being the set of outcomes of three coin flips.

Let X be the random variable "number of heads".

Then the event $(X = 2)$ is $\{HHT, HTH, THH\}$ and

$\mathbf{Pr}[X = 2] = 3/8$.

Another Example

The sample space is the set of possible inputs of length n to an algorithm.

There is the uniform distribution.

X is the time taken by the algorithm.

Average

If X is a random variable, then its average is defined as

$\mathbf{E}[X] = \sum_v v\,\mathbf{Pr}[X = v]$.

For example if we are playing a game where with prob. 1/3 we gain 2 dollars, with prob. 1/3 we gain 1 dollar and with prob. 1/3 we lose 4 dollars, then our average win is

$$\frac{1}{3} \cdot 2 + \frac{1}{3} \cdot 1 + \frac{1}{3} \cdot (-4) = -\frac{1}{3}$$

Example

The sample space is the set of possible inputs of length n to an algorithm.

There is the uniform distribution.

X is the time taken by the algorithm (a random variable).

$\mathbf{E}[X]$ is the expected time taken by the algorithm (a number).

Randomized Algorithms and Difference with Average-Case Analysis

In average-case analysis one considers a standard (non-randomized) algorithm and then considers a distribution over the inputs and computes average running time.

A randomized algorithm *makes random choices* during its execution.

For a fixed input, we consider the sample space of all random choices, and the average time over these choices.

Then we take the worst-case input for the average-case running time.

Randomized Select

ChoosePivot($A[1, \ldots, n]$) chooses a random element in the vector.

For every $k = 1, \ldots, n$ there is a $1/n$ probability of choosing the element of order statistics k.

When it happens, we have a $(k-1, n-k)$ partition.

Average Running Time

Let $ET(n)$ be the random variable corresponding to the running time of randomized select for an input of length n.

We have the relation

$$ET(n) \leq cn + \sum_{k=1}^{n} \frac{1}{n} ET(\max\{k-1, n-k\})$$

$$ET(1) = 1$$

Bound

Define $U(n) = an$ where a is a constant.

We want to prove that if we choose a large enough then $ET(n) \leq U(n)$ for every n, and so $ET(n) = O(n)$.

In particular, we choose $a = 4c$, that is $U(n) = 4cn$.

Prove by induction.

Base Case

$ET(1) = 1 < 4c = \leq U(1)$

Inductive Step

Assume $ET(n') \leq U(n')$ for $n' < n$. Want to prove that $ET(n) \leq U(n)$.

$$ET(n) \leq cn + \sum_{k=1}^{n} \frac{1}{n} ET(\max\{k-1, n-k\})$$

$$\leq cn + \frac{2}{n} \sum_{k=\lceil (n-1)/2 \rceil}^{n-1} ET(k)$$

$$\leq cn + \frac{2}{n} \sum_{k=\lceil (n-1)/2 \rceil}^{n-1} U(k)$$

$$= cn + \frac{2}{n} \sum_{k=1}^{n-1} U(k) - \frac{2}{n} \sum_{k=1}^{\lceil (n-1)/2 \rceil - 1} U(k)$$

$$= cn + \frac{2}{n} \left(a\frac{n(n-1)}{2} - a\frac{\lceil (n-1)/2 \rceil (\lceil (n-1)/2 \rceil - 1)}{2} \right)$$

$$
\begin{aligned}
&< cn + \frac{2}{n}\left(\frac{an^2}{2} - \frac{an}{2} - \frac{a(n-1)(n-3)}{8}\right) \\
&< cn + \frac{3}{4}an \\
&= an \\
&= U(n)
\end{aligned}
$$