Randome Variables and Expectation

Example: Finding the k-Smallest Element in an ordered set.

Procedure Order(S, k);
Input: A set S, an integer $k \leq |S| = n$.
Output: The k smallest element in the set S.

Example: Finding the k-Smallest Element

Procedure Order(S, k);
Input: A set S, an integer $k \leq |S| = n$.
Output: The k smallest element in the set S.

1. If $|S| = k = 1$ return S.
2. Choose a random element y uniformly from S.
3. Compare all elements of S to y. Let $S_1 = \{x \in S \mid x \leq y\}$ and $S_2 = \{x \in S \mid x > y\}$.
4. If $k \leq |S_1|$ return Order(S_1, k) else return Order($S_2, k - |S_1|$).

Theorem

1. *The algorithm always returns the k-smallest element in S*
2. *The algorithm performs $O(n)$ comparisons* **in expectation**.

Random Variable

Definition

A random variable X on a sample space Ω is a real-valued function on Ω; that is, $X : \Omega \to \mathcal{R}$. A discrete random variable is a random variable that takes on only a finite or countably infinite number of values.

Discrete random variable X and real value a: the event "$X = a$" represents the set $\{s \in \Omega : X(s) = a\}$.

$$\Pr(X = a) = \sum_{s \in \Omega : X(s) = a} \Pr(s)$$

Independence

Definition

Two random variables X and Y are independent if and only if

$$\Pr((X = x) \cap (Y = y)) \;=\; \Pr(X = x) \cdot \Pr(Y = y)$$

for all values x and y. Similarly, random variables $X_1, X_2, \ldots X_k$ are mutually independent if and only if for any subset $I \subseteq [1, k]$ and any values x_i,$i \in I$,

$$\Pr\left(\bigcap_{i \in I} X_i = x_i\right) \;=\; \prod_{i \in I} \Pr(X_i = x_i).$$

Expectation

Definition

The expectation of a discrete random variable X, denoted by $\mathbf{E}[X]$, is given by

$$\mathbf{E}[X] \;=\; \sum_i i \Pr(X = i),$$

where the summation is over all values in the range of X. The expectation is finite if $\sum_i |i| \Pr(X = i)$ converges; otherwise, the expectation is unbounded.

The expectation (or mean or average) is a weighted sum over all possible values of the random variable.

Median

Definition

The **median** of a random variable X is a value m such

$$Pr(X < m) \leq 1/2 \quad \text{and} \quad Pr(X > m) < 1/2.$$

Linearity of Expectation

Theorem

For any two random variables X and Y

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

Lemma

For any constant c and discrete random variable X,

$$\mathbf{E}[cX] = c\mathbf{E}[X].$$

Example: Finding the k-Smallest Element

Procedure Order(S, k);
Input: A set S, an integer $k \leq |S| = n$.
Output: The k smallest element in the set S.

1. If $|S| = k = 1$ return S.
2. Choose a random element y uniformly from S.
3. Compare all elements of S to y. Let $S_1 = \{x \in S \mid x \leq y\}$ and $S_2 = \{x \in S \mid x > y\}$.
4. If $k \leq |S_1|$ return Order(S_1, k) else return Order($S_2, k - |S_1|$).

Theorem

1. *The algorithm always returns the k-smallest element in S*
2. *The algorithm performs $O(n)$ comparisons* **in expectation**.

Proof

- We say that a call to Order(S, k) was *successful* if the random element was in the middle $1/3$ of the set S. A call is successful with probability $1/3$.
- After the i-th successful call the size of the set S is bounded by $n(2/3)^i$. Thus, need at most $\log_{3/2} n$ successful calls.
- Let X be the total number of comparisons. Let T_i be the number of iterations between the i-th successful call (included) and the $i+1$-th (excluded): $\mathbf{E}[X] \leq \sum_{i=0}^{\log_{3/2} n} n(2/3)^i \mathbf{E}[T_i]$.
- T_i has a geometric distribution $G(1/3)$.

The Geometric Distribution

Definition

A geometric random variable X with parameter p is given by the following probability distribution on $n = 1, 2, \ldots$.

$$\Pr(X = n) = (1 - p)^{n-1} p.$$

Example: repeatedly draw independent Bernoulli random variables with parameter $p > 0$ until we get a 1. Let X be number of trials up to and including the first 1. Then X is a geometric random variable with parameter p.

Lemma

Let X be a discrete random variable that takes on only non-negative integer values. Then

$$\mathbf{E}[X] = \sum_{i=1}^{\infty} \Pr(X \geq i).$$

Proof.

$$\begin{aligned}
\sum_{i=1}^{\infty} \Pr(X \geq i) &= \sum_{i=1}^{\infty} \sum_{j=i}^{\infty} \Pr(X = j) \\
&= \sum_{j=1}^{\infty} \sum_{i=1}^{j} \Pr(X = j) \\
&= \sum_{j=1}^{\infty} j \Pr(X = j) = \mathbf{E}[X].
\end{aligned}$$

For a geometric random variable X with parameter p,

$$\Pr(X \geq i) = \sum_{n=i}^{\infty} (1-p)^{n-1} p = (1-p)^{i-1}.$$

$$\begin{aligned} \mathbf{E}[X] &= \sum_{i=1}^{\infty} \Pr(X \geq i) \\ &= \sum_{i=1}^{\infty} (1-p)^{i-1} \\ &= \frac{1}{1-(1-p)} \\ &= \frac{1}{p} \end{aligned}$$

Proof

- Let X be the total number of comparisons.
- Let T_i be the number of iterations between the i-th successful call (included) and the $i+1$-th (excluded):
- $\mathbf{E}[X] \leq \sum_{i=0}^{\log_{3/2} n} n(2/3)^i \mathbf{E}[T_i]$.
- $T_i \sim G(1/3)$, therefore $\mathbf{E}[T_i] = 3$.
- Expected number of comparisons:

$$\mathbf{E}[X] \leq \sum_{j=0}^{\log_{3/2} n} 3n \left(\frac{2}{3}\right)^j \leq 9n.$$

Theorem

1. *The algorithm always returns the k-smallest element in S*
2. *The algorithm performs $O(n)$ comparisons* **in expectation**.

What is the probability space?

Finding the k-Smallest Element with no Randomization

Procedure Det-Order(S, k);
Input: An array S, an integer $k \leq |S| = n$.
Output: The k smallest element in the set S.

1. If $|S| = k = 1$ return S.
2. Let y be the first element is S.
3. Compare all elements of S to y. Let $S_1 = \{x \in S \mid x \leq y\}$ and $S_2 = \{x \in S \mid x > y\}$.
4. If $k \leq |S_1|$ return Det-Order(S_1, k) else return Det-Order$(S_2, k - |S_1|)$.

Theorem

The algorithm returns the k-smallest element in S and performs $O(n)$ comparisons **in expectation** *over all possible input permutations.*

Randomized Algorithms:

- Analysis is true for **any** input.
- The sample space is the space of random choices made by the algorithm.
- Repeated runs are independent.

Probabilistic Analysis:

- The sample space is the space of all possible inputs.
- If the algorithm is **deterministic** repeated runs give the same output.

Algorithm classification

A **Monte Carlo Algorithm** is a randomized algorithm that may produce an incorrect solution.
For decision problems: A **one-side error** Monte Carlo algorithm errs only one one possible output, otherwise it is a **two-side error** algorithm.
A **Las Vegas** algorithm is a randomized algorithm that **always** produces the correct output.
In both types of algorithms the run-time is a random variable.