Оглавление

Лекция 6: Схема Бернулли, схема Уолкера, неравенство Крафта.

18.10.2023

0.1 Схема Бернулли

Определение 1. Пусть $\delta_1, ..., \delta_n$ — последовательность независимых одинаково распределенных случайных величин, каждая из которых принимает значение 1 с вероятностью p и значение 0 с вероятностью $q = 1 - p$. Такая вероятностная схема называется схемой Бернулли.

Определение 2. Случайная величина $\xi_n = \delta_1 + \ldots, \delta_n$ имеет биномиальное распределение:

$$p(\xi_n = k) = C_n^k p^k q^{n-k}$$

Замечание. Чтобы найти $\xi_n = k$ нужно чтобы k из случайных величин $\delta_1, ..., \delta_n$ принимали значение 1, остальные - 0. Вероятность этого события при фиксированных местах единиц и нулей равна $p^k q^{n-k}$, и если учесть все возможные C_n^k расположений этих мест, то получим $k-$ый член биномиального разложения $(p + q)^n$

Свойства. Т.к. $E_{\xi_1} = p$ и $D_{\xi_1} = p - p^2 = pq$:

1. $E_\xi = np$
2. $D_\xi = npq$

0.2 Случайные числа и схема Уолкера

Дискретная случайная величина — это такая случайная величина, значения которой могут быть не более чем счетными, то есть либо конечными,

либо счетными. Под счётностью имеется ввиду, что значения случайной величины можно занумеровать.

В вычислительных машинах можно имитировать случайные эксперименты. В качестве источника случайности используются специальные программы - *датчики случайных чисел*. Датчик при каждом обращении к нему вырабатывает некоторое число (обычно целое или вещественное число из фиксированного диапазона), и последовательность этих чисел по своему поведению очень похожа на последовательность независимых *случайных величин, имеющих одинаковое равномерное распределение.*

Есть стандартные функции — генераторы (псевдо-)случайных чисел, которые выдают случайные натуральные числа в диапазоне $[0 : n]$, либо вещественные из $[0 : 1)$. Равномерное распределение обладает таким свойством, что вероятность того, что случайная величина принимает значения из некоторого множества, пропорциональна количеству элементов в нем. Или же что вероятность попадания в промежуток $[a, b] \subset [0 : 1)$ равна длине этого промежутка.

Определение 3. Равномерное (дискретное) распределение — распределение на конечном множестве, в котором все исходы равновероятны.

Пример. Равномерное распределение на множестве целых чисел от k до l :

$$p(\xi = S) = \frac{1}{l - k + 1}, S \in [k : l]$$

Алгоритм (Схема Уолкера). Пусть мы умеем получать случайное число из диапазона $[0 : 1)$. Требуется смоделировать вероятностную схему из m исходов с заданными вероятностями $p_1, p_2, ..., p_m$.

- Назовём “донорами” те исходы, которые получили больше, чем им нужно, т.е. $p_i < \frac{1}{m}$.
- Назовём “реципиентами” те исходы, которые получили меньше, чем им нужно, т.е. $p_i > \frac{1}{m}$.

Алгоритм перераспределения:

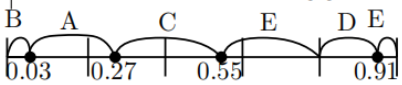
1. Берём произвольного донора и реципиента.
2. Донор отдаёт реципиенту излишек из своего отрезка.
3. Отмечаем точку, где донор отдал излишек реципиенту – барьер.
4. После этого у донора остаётся ровно столько, сколько ему нужно, а реципиент мог как остаться реципиентом (ему дали слишком мало), так и перейти в доноры (дали с избытком).

И так до тех пор, пока доноры и реципиенты не закончатся. После проведения такой предварительной работы можно за $O(1)$ генерировать случайные числа с нужным нам распределением следующим образом:

1. Генерируем случайное число x из диапазона $[0:1)$.
2. берем $\lfloor xm \rfloor$ – номер интервала на отрезке.
3. Сравниваем x с барьером на этом отрезке: если x больше, возвращаем реципиент, если меньше – донора.

Пример:
$m = 5, p(A) = 0.24, p(B) = 0.03, p(C) = 0.28, p(D) = 0.11, p(E) = 0.34$

i	Донор	Реципиент	Барьер
1	B	A	0.03
2	A	C	0.27
3	C	E	0.55
4	D	E	0.91



0.3 Двоичный поиск и неравенство Крафта.

Замечание. Данная глава писалась с опорой на другие источники. Думаю, вы понимаете почему. ⊙ ⌣ ⊙

Определение 4. Алфавит A – конечное непустое множество символов, $\alpha \in A^k$ – строка длины k над алфавитом A.

Определение 5. $A^* = \bigcup_{k=0}^{\infty} A^k$ – множество всех строк над алфавитом A (слова). ε – пустая строка.

Определение 6. Кодом называется функция $\varphi : A^* \to B^*$. B – алфавит кода.

Определение 7. Код называется декодируемым, если $\forall \alpha, \beta \in A^* : \alpha \neq \beta \Rightarrow \varphi(\alpha) \neq \varphi(\beta)$, т.е. φ – инъекция.

Пример. Операция сжатия каким либо архиватором – декодируемый код. *jpeg* – сжатие с потерей информации, недекодируемый код.

Определение 8. Разделяемый код – код, в котором каждый символ алфавита A кодируется отдельно, т.е. код – функция $\varphi : A \to B^*$.

Если хотим закодировать строку $\alpha = a_1 a_2 ... a_n$, то $\varphi(\alpha) = \varphi(a_1)\varphi(a_2)...\varphi(a_n)$ (конкатенируем коды символов)

Определение 9. Префиксный код – функция $\varphi : A \to B^*$, такая, что $\forall a, b \in A : \varphi(a)$ – не префикс $\varphi(b)$, т.е. ни одно кодовое слово не является префиксом другого кодового слова.

Пример. Пример: $A = \{a, b, c\}, B = \{0, 1\}$
$\varphi(a) = 0, \varphi(b) = 10, \varphi(c) = 11 \Rightarrow \varphi$ – не префиксный код.

Теорема 1. Если код префиксный, то он однозначно декодируемый.

Доказательство. Приведем алгоритм декодирования.

Пусть дана $t = \varphi(s)$, нужно найти s. Тогда будем из t выделять префикс (он будет один, т.к. код префиксный), который является кодом некоторого символа a_1. Отрежем этот префикс от t. Так делаем, пока t не кончится. Получим $a_1 a_2 ... a_n = s$. □

Теорема 2 (неравенство Крафта). Для алфавита $A = \{c_1, ..., c_k\}$ можно построить однозначно декодируемый двоичный код с длинами кодовых слов $s_1, \ldots, s_k$ если:

$$\sum_{i=1}^{k} 2^{-s_i} \leq 1 \qquad 2$$

И наоборот, если для чисел $s_1, \ldots, s_k$ выполняется неравенство 2, то существует однозначно декодируемый двоичный код в алфавите A.

Доказательство. (Необходимость) Рассмотрим бинарное дерево T. Каждой вершине r сопоставим число $a_r = 2^{-l}$, где l длина пути от корня до вершины r. Для корня имеем $a_{root} = 1$.
Заметим, что для любого узла, который не лист, справедливо неравенство:

$$2^{-l} \geq \underbrace{2^{-(l+1)}}_{left} + \underbrace{2^{-(l+1)}}_{right}$$

Просуммируем все узлы, которые не являются листьями и все узлы не корни, тогда справедливо неравенство:

$$\sum_{nodes \setminus leaves} a_r \geq \sum_{nodes \setminus \{a_{root}\}} a_r$$

После сокращения общих слагаемых в левой и правой частях неравенства получаем корень слева и все листья справа:

$$a_{root} = 1 \geq \sum_{leaves} a_r = \sum_{i=1}^{k} 2^{-s_i}$$

□