Санкт-Петербургский государственный университет
www.spbu.ru

Билеты по дискретной математике

1-2 семестр, преподаватель Григорьева Н. С.[1]
Записали Костин П.А. и Щукин И.В.[2]

[1]Также были использованы учебник Романовский И.В. "Дискретный анализ" и интернет

[2]Данный документ неидеальный, прошу сообщать о найденных недочетах вконтакте (можно присылать скрины неточностей с указанием билетов)

Содержание

Некоторые определения из теории множеств. Прямое произведение, разбиение множеств. Мощность объединения

Опр

Пустое множество ($\varnothing$) - мно-во, которому $\notin$ ни один элемент

Опр

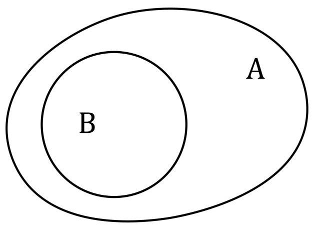
Число элементов мн-ва A - мощность $|A|$

Опр

Множество чисел от k до l обозначается $k : l$

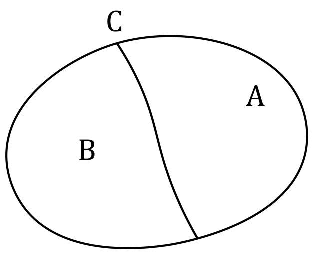
Опр

Мн-во A - подмн-во мн-ва B ($A \subset B$), если каждый элемент из A принадлежит B



Опр

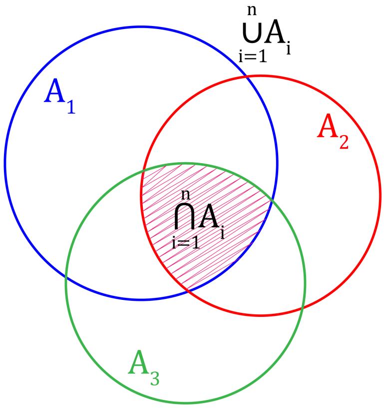
C - объединение A и B ($A \cup B$), если оно состоит из всех элементов A и B ($C = \{x | x \in A$ и $x \in B\}$)



Опр

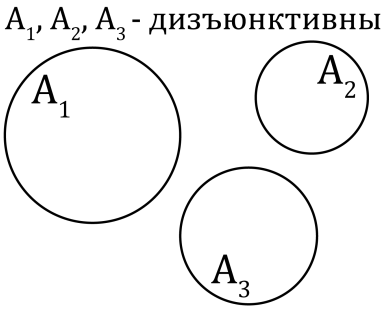
$\bigcup\limits_{i=1}^{n} A_i, \quad \bigcap\limits_{i=1}^{n} A_i$ - объединение и пересечение конечного числа мн-в

$(\bigcup\limits_{i \in I} A_i, \quad \bigcap\limits_{i \in I} A_i)$ - аналогично



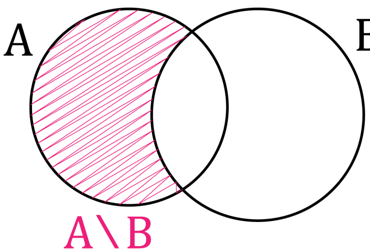
Опр

Если пересечение мн-в пусто, то они называются дизъюнктивными



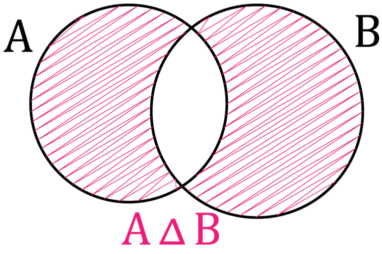
Опр

Мн-во С называется разностью мн-в А и В ($C = A \backslash B$), если оно состоит из всех эл-в, принадлежащих А и не принадлежащих В



Опр

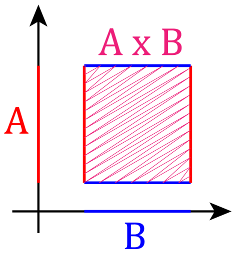
$A \triangle B = A \setminus B \cup B \setminus A$ - симметрическая разность



Опр

Мн-во упорядоченных пар (i, j), где $i \in A$, $j \in B$ называется прямым произведением мн-в A и B

$$A \times B = \{(i, j) \mid i \in A, \quad j \in B\}$$

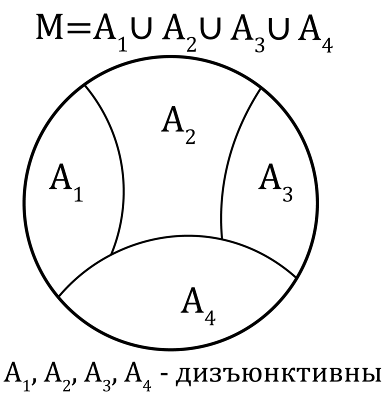


Замечание

Мощность прямого произведения $|A \times B| = |A| \cdot |B|$. Аналогично произведение $\forall$ конечного числа множеств

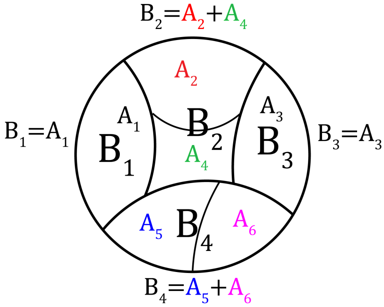
Опр

Пусть $A_1, ..., A_k$ - ненулевые и попарно дизъюнктивные, $M = A_1 \cup ... \cup A_k$, тогда мн-во $\{A_1, ..., A_k\}$ называется разбиением M
(если они попарно не дизъюнктивные, тогда это покрытие)



Опр

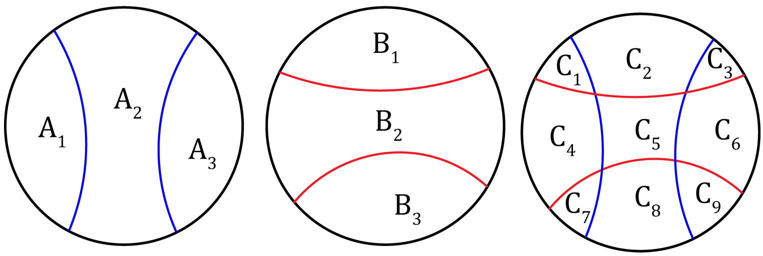
Разбиение A мн-ва M называется измельчением B, если $\forall A_i \in A$ содержится в некотором $B_i \in B$



A - измельчение B

Опр

Пусть A, B - разбиения мн-ва M, разбиение C называется произведением A и B, если оно является их измельчением, причем самым крупным $C = A \cdot B$



C - произведение разбиений A и B

Теорема

Произведение двух разбиений существует

Док-во

Предъявим разбиение, которое будет пересечением $A = \{A_1, ..., A_k\}$ и $B = \{B_1, ..., B_l\}$, точнее $D_{ij} = A_i \cap B_j, \quad i \leqslant k, \quad j \leqslant l$

$$\Rightarrow \mathcal{P} = \cup D_{ij} \text{ (т.е. без пустых строк)}$$

Покажем, что тогда оно самое крупное

Пусть $\exists F = \{F_1, ..., F_t\}$ - измельчение A и B, тогда:

$$\forall F_k \quad \exists A_{i_k},\ B_{i_k} : F_k \subset A_{i_k},\ B_{i_k} \Rightarrow F_k \subset (A_{i_k} \cup B_{i_k}) = D_{i_k j_k} \Rightarrow \text{ мельче F}$$

Вектора из нулей и единиц Пусть мн-во B состоит из двух элементов которые отождествляются с 0 и 1, т.е. $B = 0 : 1$

Произведение m экзмемпляров такого мн-ва обозначим за $B^m = (0 : 1)^m$, состоит из 2^m эл-ов

Опр

Вектор из нулей и единиц - упорядоченный набор из фиксированного числа нулей и единиц, т.е. эл-т мн-ва B^m

Упорядоченный набор из чисел обычно называется вектором, m - размерностью вектора, каждый отдельный элемент набора - компонента вектора

Замечание

Модели, в которых используются наборы из 0 и 1:

1. Геометрическая интерпретация

   Точкой в m-мерном пространстве является m-мерный вектор, каждая его компонента - одна из декартовых координат точки. Набор из 0 и 1, рассматриваемый как точка в пространстве, определяет вершину куба, построенного на ортах (единичных отрезках) координатных вероятностей

2. Логическая интерпретация

   Операции над векторами выполняются покомпонентно, т.е. независимо над соотв. компонентами векторов-операндов

Пример

$$\begin{array}{c|ccccc} x & 0 & 0 & 0 & 1 & 1 \\ y & 1 & 1 & 1 & 0 & 1 \\ x \wedge y & 0 & 0 & 0 & 0 & 1 \\ x \vee y & 1 & 1 & 1 & 1 & 1 \\ x \equiv y & 0 & 0 & 0 & 0 & 1 \\ x \not\equiv y & 1 & 1 & 1 & 1 & 0 \end{array}$$

3. Двоичное представление (натуральные числа)

   Число представляется в виде суммы степеней 2

4. Состояние памяти компьютера

5. Сообщение, передаваемое по каналу связи

6. Можно задавать подмножества мн-ва $1:n$

Алгоритм перебора 0-1 векторов. Коды Грея

Опр

Код Грея — такое упорядочение k-ичных (обычно двоичных) векторов, что соседние вектора отличаются только в одном разряде

Алгоритм

it - номер итерации, k_{it} - номер обновляемой компоненты

x$_4$	x_3	x_2	x_1	it	k$_{it}$
0	0	0	0	0	1
0	0	0	1	1	2
0	0	1	1	2	1
0	0	1	0	3	3
0	1	1	0	4	1
0	1	1	1	5	2
0	1	0	1	6	1
0	1	0	0	7	4
	...			...	

Суть алгоритма: зафиксируем нулевое значение у m-й компоненты и переберем все наборы длины $m-1$ для ост. компонент. Перебрав их меняем значение m-й компоненты на 1 и перебинаем набор длины $m-1$ в обратном порядке

Замечание*

Явная формула для проверки $G_i = i \oplus (\lfloor i/2 \rfloor)$

Перебор элементов прямого произведения множеств

$$M(1:k) = M_1 \times M_2 \times ... \times M_k$$

$$|M_1 \times M_2 \times ... \times M_k| = \prod_{i \in 1:k} m_i, \text{ где } m_i = |M_i|$$

Пусть каждое M_i состоит из целых чисел от 0 до $m_i - 1$, тогда каждый элемент $M(1:k)$ - последовательность неотрицательных чисел $r_1, ..., r_k$, причем $r_i < m_i$

$$\text{num}(r_1, ..., r_k) = \sum_{i=1}^{k} r_i \cdot (\prod_{j=1}^{i-1} m_j) = r_1 + r_2 m_1 + ... + r_k m_1 \cdot ... \cdot m_{k-1}$$

Размещения, сочетания, перестановки без повторений

Опр

Перестановка из n без повторений - упорядоченный набор из n неповторяющихся элементов, каждый из которых берется из диапазона $1:n$

$$P_k = n!$$

Опр

Размещение - упорядоченный набор из k неповторяющихся элементов из диапазона $1:n$

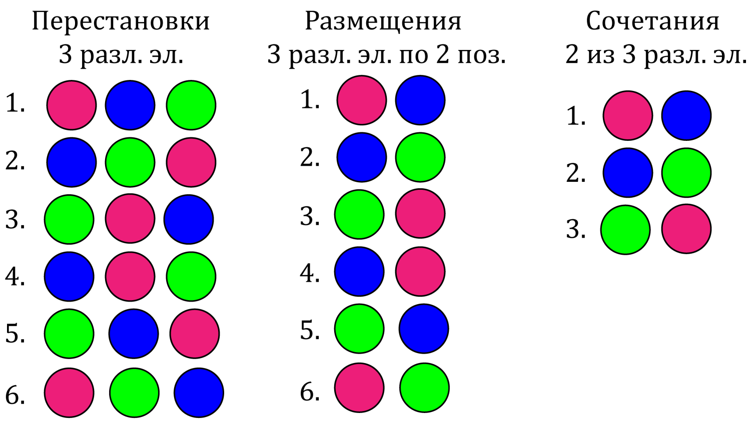
$$A_n^k = \frac{n!}{(n-k)!} = n(n-1)(n-k+1)$$

Или $A_n^k = C_n^k k! = \dfrac{n!\cdot k!}{(n-k)!\cdot k!}$

Опр

Сочетание - набор из k неповторяющихся элементов из диапазона $1:n$ (порядок не важен)

$$C_n^k = \frac{A_n^k}{P_k} = \frac{n!}{(n-k)!k!}$$



Размещения, сочетания, перестановки с повторениями

Опр

Перестановка - последовательность длины n, составленная из k разных символов, i-ый из которых повторяется n_i раз $(n_1 + n_2 + ... + n_k = n)$

$$P(n_1, n_2, ..., n_k) = \frac{n!}{n_1!\cdot ...\cdot n_k!}$$

Пример **(Перестановки с повторениями, aabc)**

Перестановки:

$$abac,\ baac,\ aabc,\ aacb,\ abca,\ baca,\ acba,\ acab,\ bcaa,\ cbaa,\ caba,\ caab$$

Опр **(размещения с повторениями)**

Аналогичное определение

$$\overline{A}_n^k = n^k$$

Опр **(сочетания с повторениями)**

Аналогичное определение

$$|\overline{C}_n^k| = C_{n+k-1}^k = C_{n+k-1}^{n-1}$$

Два алгоритма перебора перестановок. Нумерация перестановок

$$|P_k| = k! = |T_k|$$

P_k - мн-во всех перестановок

$$T_k = \prod_{i=1}^{k} M_i = \{0\} \times \{0,1\} \times ... \times \{0,1,...,k-1\}$$

Построим взаимно однозначное соответствие между P_k и T_k. Возьмем перестановку $(t_1, ..., t_k)$ и сопоставим ей перестановку $(r_1, ..., r_k)$ следующим образом: для любого $i \in 1 : k$ найдем число значений, меньше r_i среди $r_{i+1}, ..., r_k$ - это число мы и примем в качестве t_i

В соответсвии с таким определеничем чисел t_i в мн-ве T_k будет естественно сделать значения m_i не возрастающими, а убывающая до единицы

Разберем $r = (4,\ 8,\ 1,\ 5,\ 7,\ 2,\ 3,\ 6)$

i	1	2	3	4	5	6	7	8
r_i	4	8	1	5	7	2	3	6
t_i	3	6	0	2	3	0	0	0
m_i	8	7	6	5	4	3	2	1

$$t_1 = |\{1,2,3\}| \quad t_2 = |\{1,5,7,2,3,6\}|, \quad t_3 = |\{\}|, \quad ...$$

По $(t_1, ..., t_k)$ легко восстановить исходную перестановку. Для этого меняя i от 1 до k нужно нужно проверить мн-во значений S_i, которые могут быть в перестановке на i месте.

В нашем примере для $i = 1$ $S_1 = 1 : 8$, $t_1 = 3 \Rightarrow r_1 = 4$, далее $S_2 = 1 : 3 \cap 5 : 8$, $t_2 = 6 \Rightarrow r_2 = 8$. Если использовать это отображение при переборе, то перестановки будут перебираться в лексикографическом порядке

Опр

$(r_1, ..., r_k)$ предшествует $(R_1, ..., R_k)$, если начала перестановок совпадают до индекса i, а дальше $r_i < R_i$

Алгоритм (1)

Если перестановки перебираются в лексикографическом порядке, можно вывести правило получения следующего:

1. В перестановке $(r_1, ..., r_k)$ найти наибольший суффикс $(r_t, ..., r_k)$, в котором элементы расположены по убыванию $r_t > ... > r_k$; ($r_{t-1} < r_t$ - суффикс максимальн)

2. Выбрать $(r_t, ..., r_k)$ элемент следующий по велечине после r_{t-1} и поставить его на место $t - 1$. Оставшиеся элементы, включая r_{t-1} расположить в порядке возрастания

num	t$_k$				p$_k$			
0	0	0	0	0	1	2	3	$\underline{4}$
1	0	0	1	0	1	2	$\underline{4}$	$\underline{3}$
2	0	1	0	0	1	3	2	$\underline{4}$
3	0	1	1	0	1	3	$\underline{4}$	$\underline{2}$
4	0	2	0	0	1	4	2	$\underline{3}$
5	0	2	1	0	1	$\underline{4}$	$\underline{3}$	$\underline{2}$

Замечание

Чтобы найти номер перестановки используем факториальную запись:

$$\begin{array}{lcccc} p & 3 & 4 & 2 & 1 \\ t & \underset{3}{2} & \underset{2}{2} & \underset{1}{1} & \underset{0}{0} \end{array}$$

$$\text{num} = 2 \cdot 3! + 2 \cdot 2! + 1 \cdot 1! + 0 \cdot 0! = 7$$

Так можем, например, найти перестановку через n шагов от данной: сначала ищем номер исходной, прибавляем n, затем выполняя поэтапное деление в столбик на значение факториалов, восстанавливаем перестановку

Алгоритм (2)

$r = (1, 2, ..., k)$ - рабочая перестановка

$t = (0, 0, ..., 0)$ - номер r в факториальной системе счисления

(младший разряд последний)

$d = (-1, -1, ..., -1)$ - направление движения элементов

$p = (1, 2, ..., k)$ - сопоставление каждому i места, в котором он в r

1. Увеличить t на 1. При этом несколько младших разрядов получат нулевые значения, в j-м значении ув-ся на 1. При $j = 1$ процесс заканчивается

2. Сменить направление движения всех элементов младше j-го ($d_i = -d_i$ для $i > j$). Поменять местами j и соседний с ним (если $d_j - -1$ - левый, d_j - правый)

i	t	d	p	r	j	Комментарий
1	0000	- - - -	1234	1234	-	
2	0001	- - - -	1243	1243	4	Нач-ся движение эл-та 4
4	0003	- - - -	2341	4123	4	
5	0010	- - - +	2431	4132	3	Шаг эл-та 3, у 4 смена направления
6	0011	- - - +	1432	1432	4	
7	0012	- - - +	1423	1342	4	
8	0013	- - - +	1324	1324	4	
9	0020	- - - -	2314	3124	3	Второй шаг эл-та 3

Задача о минимуме скалярного произведения Пусть заданы числа $x_1, ..., x_m$ и $y_1, ..., y_m$. Составим пары (x, y), включив каждое x_i и y_i ровно в одну пару. Затем перемножим числа каждой пары и сложим полученное произведение. Требуется найти min такое разбиение чисел на пары S

Теорема

$$\overline{x} = (x_1, ..., x_n) \qquad x_1 \geqslant x_2 ... \geqslant x_n$$

$$\overline{y} = (y_1, ..., y_n) \qquad y_1 \leqslant y_2 ... \leqslant y_n$$

$$S = \sum_{i=1}^{n} x_i y_i \to \min$$

Док-во

Покажем, что если найдутся пары чисел $(x_i,\ y_i)$ и $(x_j,\ y_j)$: $x_i < x_j$, $y_i < y_j$, то S можно уменьшить, заменив парами $(x_i,\ y_j)$ и $(x_j,\ y_i)$

Действительно, так как $(x_j - x_i)(y_j - y_i) > 0$, то, раскрывая скобки, получим после переноса $x_i y_i + x_j y_j > x_i y_j + x_j y_i$

Поскольку число возможных расположений равно m!, т.е. конечное число, то начиная с любого расположения за конечное число шагов мы закончим процесс улучшений на расположении, которое дальше улучшить невозможно. Но нем и достигается минимум

Числа Фибоначчи. Теорема о представлении

Опр

Последовательность чисел Фибоначчи F:

$$F_0 = 0, \quad F_1 = 1, \quad F_n = F_{n-1} + F_{n-2},\ n > 1$$

Утв

$$\varphi_n = \frac{F_{n+1}}{F_n} \text{ - сходится}$$

Следствие

$$\varphi_n = \frac{F_{n+1}}{F_n} = \frac{F_{n-1} + F_n}{F_n} = 1 + \frac{1}{\varphi}$$

$$\Rightarrow \varphi = \frac{\sqrt{5} + 1}{2}$$

Лемма

При $n > 1$ выполнено $\varphi^{n+2} = \varphi^{n+1} + \varphi^n$

Док-во

Лемма

При $k > 2$ выполнено:

$$F_{2k} = F_{2k-1} + F_{2k-3} + ... + F_1$$

$$F_{2k+1} = 1 + F_{2k} + F_{2k-2} + ... + F_0$$

Док-во (по индукции)

$(k=3)$:

$$F_6 = 8 = 5 + 2 + 1$$

$$F_7 = 13 = 1 + 8 + 3 + 1 + 0$$

$(k \to k+1)$:

$$F_{2(k+1)} = F_{2k+2} = F2k + 1 + F_{2k} = F_{2k+1} + F_{2k-1} + ... + F_1 = ???$$

Теорема

Любое натуральное число можно однозначно представить в виде суммы чисел Фибоначчи

$$s = F_{i_0} + F_{i_1} + ... + F_{i+r}, \text{ где } i_{k-1} + 1 < i_k, \quad k \in 1 : r \quad i_0 = 0$$

Док-во

Существование:

Пусть $j(s)$ - номер масимального числа Фиббоначи, не превосходящего s. Положим $s' = s - F_j(s)$. Из определения $j(s)$ следует, что $s' < F_{j(s)-1}$, иначе число Фиббоначи не было бы максимальным. Теперь мы получим искобое представление для s как представление s', дополненное слагаемым $F_{j(s)}$

Единственность:

Пусть есть ещё одно представление $s = F_{j_0} + ... + F_{j_q}$. Н.У.О. считаем, что $j_q < j(s)$. Если мы заменим F_{j_q} на $F_{j(q)-1}$, то правая часть разве что лишь увеличится. Аналогично заменим с возможным увеличением предпоследнее слагаемое на $F_{j(s)-3}$. ???

Перебор сочетаний. Нумерация сочетаний Состояние вычислительного процесса. Массив $(x_1, ..., x_m)$ номеров, включенных в сочетание. Начальное состояние: принять $x_i = i \quad \forall i \in 1 : m$. Стандартный шаг: просматривать компоненты вектора x, начиная с x_m и искать первую компоненту, которую можно увеличить (нельзя $x_m = n$, $x_{m-1} = n - 1$ и т.д.). Если такой нет, то закончить процесс. В противном случае пусть k - наибольшее число, для которого $x_k < n - m + k$, тогда увеличиьть x на единицу, а для всех следующиъ за k-ый продолжаем, но ряд от значения x_k, т.е. $x_i = x_k + (i - k)$

num	Сочетание					k
1	1	2	3	4	5	5
2	1	2	3	4	6	5
3	1	2	3	4	7	5
4	1	2	3	5	6	4
5	1	2	3	5	7	5

Удобно использовать вектора из 0 и 1, чтобы перенумеровать. С каждым сочетанием из n по m можно связать вектор из n нулей и единиц, в криром единиц ровно m - числа, входящие в данное сочетание просто задают номера этих единиц

$$\mathrm{num}(b[1:n],\ m) = \begin{cases} \mathrm{num}(b[1:n],\ n) & b[n]=0 \\ l_{n-1}^{m} + \mathrm{num}(b[1:n],\ m-1) & b[n]=1 \end{cases}$$

Пример

$$\mathrm{num}((0,\ 1,\ 0,\ 1,\ 0,\ 0,\ 1),\ 3) =$$
$$= C_6^3 + \mathrm{num}((0,\ 1,\ 0,\ 1,\ 0,\ 0),\ 2) = C_6^3 + C_3^2 + \mathrm{num}((0,\ 1,\ 0),\ 1) =$$
$$= C_6^3 + C_3^2 + \mathrm{num}((0,\ 1),\ 1) = C_6^3 + C_3^2 + C_1^1 + \mathrm{num}((0),\ 0) = 24$$

Бином Ньютона и его комбинаторное использование Треугольник Паскаля (в узлах C_n^k):

$$\begin{array}{ll} & \qquad\qquad\qquad 1 \\ n=0 & \qquad\qquad\quad 1 \quad 1 \\ n=1 & \qquad\qquad 1 \quad 2 \quad 1 \\ n=2 & \qquad\quad 1 \quad 3 \quad 3 \quad 1 \\ n=3 & \qquad 1 \quad 4 \quad 6 \quad 4 \quad 1 \\ n=4 & \quad 1 \quad 5 \quad 10 \quad 10 \quad 5 \quad 1 \\ n=5 & \underline{1 \quad 6 \quad 15 \quad 20 \quad 15 \quad 6 \quad 1} \\ n=6 & 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \end{array}$$

Опр

Бином Ньютона: $(a+b)^n = \sum_{k=0}^{n} C_n^k a^k b^{n-k}$

Лемма

$$C_{n-1}^{k-1} + C_{n-1}^{k} = C_n^k$$

Док-во **(по индукции)**

$$(\mathrm{a+b})^n = a(a+b)^{n-1} + b(a+b)^{n-1} =$$
$$= \sum_{k=0}^{n-1} C_{n-1}^{k} a^{k+1} b^{(n-1)-k} + \sum_{k=0}^{n-1} C_{n-1}^{k} a^{k} b^{1+(n-1)-k} =$$
$$= \sum_{k=1}^{n} C_{n-1}^{k-1} a^{k} b^{n-k} + \sum_{k=0}^{n-1} C_{n-1}^{k} a^{k} b^{n-k} = \sum_{k=0}^{n} (C_{n-1}^{k-1} + C_{n-1}^{k}) a^{k} b^{n-k}$$

Следствие

$a = 1, \quad b = 1$:

$$\sum_{k=0}^{n} C_n^k = 2^n$$

$a = 1, \quad b = -1$:

$$\sum_{k=0}^{n} C_n^k (-1)^k = 0$$

(благодаря $C_n^k = C_n^{n-k}$)

$a = 1, \quad b = i$:

$$\sum_{k=0}^{n} C_n^k i^k = (1+i)^n = (\sqrt{2} \cdot \cos\frac{\pi}{4} + i \sin\frac{\pi}{4})^n = 2^{\frac{n}{2}} \cdot e^{in\frac{\pi}{4}}$$

Свойства биномиальных коэффициентов Определения см. в прошлом билете

1. $C_n^k = C_n^{n-k}$
2. $C_{n-1}^{k-1} + C_{n-1}^k = C_n^k$
3. $C_n^m C_m^k = C_n^k C_{n-k}^{m-k}$

Основные определения теории вероятностей

Опр

A - событие, $p(A) \subset [0;1]$ - характеристика события, p - вероятность события A

Опр

S - мн-во элементарных событий (мн-во исходов), если его элементы равноправны

Опр

Пусть A - событие, $A \subset S$, тогда:

$$\frac{|A|}{|S|} = p(A)$$

Событие с вероятностью 1 называется достоверным, событие с вероятностью 0 - невозможным

Опр (совмещение событий)

Событие, которое составлено из всех элементарных событий (исходов), входящих и в А, и в В, называется совмещением А и В ($A \cup B$)

$$P(A \cup B) \leqslant P(A),\ P(B)$$

Опр

Событие, состоящее из всех эл. событий, вхолдящих или в А, или в В, называется объединением событий А и В ($A \cap B$)

Опр

А, В - события, $P(A \cup B) = 0$ (события, которые не могут вместе выполниться) $=$ А, В - несовместные события

$$P(A) + P(B) = P(A \cap B)$$

Опр

События А и В называются независимыми, если $P(A \cup B) = P(A) \cdot P(B)$

Опр

Разбиение мн-ва S на несовместные события $S_1, ..., S_n$ называется полной системой событий

$$P(A) = \sum_{i=1}^{m} P(A \cup S_i) \text{ - ф-ла полной вероятности}$$

Опр

$A_1, ..., A_k$ - независимы в совокупности, если:

$$\forall I \subset 1:k \quad P(\underset{i \in I}{\cup} A_i) = \prod_{i \in I} P(A_i)$$

Замечание

Независимость в совокупности отличается от попарной независимости, первое жестче

Пример (С.Н.Бернштейн)

Рассмотрим игральную кость в форме правильного тэтраэдра. Одна грань этой кости окрашена в белый цвет W, вторая - в черный B, третья - в красный R, а окраска четвертой - смешанная M, в ней есть все три цвета. При каждом бросании кость ложится какой-то стороной вниз. Вероятность того что на нижней грани окажется белый цвет - очевидно

$\frac{1}{2}$ (2/4). Аналогично для любого другого цвета. Вероятность выпадения двух цветов сразу - $\frac{1}{4}$ (M)

$$P(R \cup B \cup W) = \frac{1}{4} \neq P(R) \cdot P(B) \cdot P(W) = \frac{1}{8}$$

Условные вероятности и формула Байеса

Опр (Условная вероятность)

Пусть A - событие, $P(A) > 0$, тогда:

$$P(B \mid A) = \frac{P(B \cap A)}{P(A)}$$

Вероятность события B, если произошло A
(все исходы, при которых произошли B и A на исходы с A)

Замечание

События независимы, если $P(B \mid A) = P(B)$ (очевидно)

Напоминание (Формула полной вероятности)

$$B_1, ..., B_n \qquad B_i \cap B_j = \varnothing$$

$$P(A) = \sum_{i=1}^{n} P(A \cap B_i) = \sum_{i=1}^{n} P(A \mid B_i) \cdot P(B_i)$$

Пример

Старая линия завода выпускает в 2 раза меньше продукции, чем новая ($2P(S) = P(N)$). А доля брака у нее в 4 раза больше($P(Br|S) = 4P(Br|N)$). Что можно сказать о доле брака ($P(Br)$) в продукции?

Док-во

По ф-ле полной вероятности:

$$P(Br) = P(Br|S)P(S) + P(Br|N)P(N) = 4P(Br|N)P(S) + 2P(Br|N)P(S) = 2P($$

Теорема (Формула Байеса)

$$P(B_i \mid A) = \frac{P(A \mid B_i)P(B_i)}{\sum\limits_{i=1}^{n} P(A \mid B_i) \cdot P(B_i)} \qquad \text{по Григорьевой}$$

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)} \qquad \text{по Интернету}$$

Док-во

Подставим $P(B \cup A) = P(B \mid A)P(A)$ в формулу полной вероятности:

$$P(B) = \sum_i P(B \mid A_i)P(A_i)$$

$$P(B \cup A) = P(B \mid A)P(A) = P(A \mid B)P(B)$$

$$\Rightarrow P(A_i \mid B) = \frac{P(B \mid A_i)P(A_i)}{\sum_j P(B \mid A_j)P(A_j)}$$

Пример

Пусть у нас есть две колоды: 36 и 52 карты. Выбираем с равной вероятностью одну из колод. Достаем из нее карту и хотим угадать, какая это из колод. Пусть $T\diamondsuit$

$$P(B_{36}|T\diamondsuit) = \frac{P(T\diamondsuit|B_{36})P(B_{36})}{P(T\diamondsuit|B_{36})P(B_{36}) + P(T\diamondsuit|B_{52})(B_{52})} = \frac{\frac{1}{36}\frac{1}{2}}{\frac{1}{36}\frac{1}{2} + \frac{1}{52}\frac{1}{2}} = \frac{52}{88}$$

Математическое ожидание и дисперсия случайной величины

Опр

Случайная величина - числовая функция $\alpha : U \to \mathbb{R}$ на вероятностном пр-ве.

hint: U - мн-во событий

Опр

$$E(\alpha) = \sum_{u \in U} \alpha(u)\Pr(u) \text{ - мат. ожидание случайной величины } \alpha$$

$\Pr(u)$ - условная вероятность события u

Свойства

1. Если $\Pr(u) = 1 \Rightarrow E(\alpha) = \alpha(u)$
2. α, β - случ. вел. $E(\alpha + \beta) = E(\alpha) + E(\beta)$ (Линейность)
3. Если $\alpha = c\beta$, где $c = const$ $E(\alpha) = cE(\beta)$
4. $E(\alpha \cdot \beta) = E(\alpha) \cdot E(\beta)$ если α и β нез.

Док-во

$$\text{Линейность} \qquad E(\alpha+\beta)=\sum_{u\in U}(\alpha(u)+\beta(u))\Pr(u)=\sum_{u\in U}\alpha(u)\Pr(u)+$$

$$+\sum_{u\in U}\beta(u)\Pr(u)=E(\alpha)+E(\beta)$$

Опр

Мат. ожидание квадрата отклонения случайной величины от ее мат. ожидания называется дисперсией этой случайно величины

$$D(\alpha)=E(\alpha-E(\alpha))^2$$

Дисперсия характеризует разброс случайной величины вокруг ее мат. ожидания

$$D(\alpha)=E(\alpha-E(\alpha))^2=E(\alpha^2)-E(2\alpha E(\alpha))+E(E^2(\alpha))=$$

$$=E(\alpha^2)-2E(\alpha)E(E(\alpha))+E^2(\alpha)=E(\alpha^2)-E^2(\alpha)$$

Свойства (Дисперсии)

1. Дисперсия неотрицательна. Если $\Pr(u)=1$, то $D(\alpha)=0$
2. $\alpha=\beta+c, \qquad c=const, \qquad \Rightarrow D(\alpha)=D(\beta)$
3. $\alpha=c\beta, \qquad c=const, \qquad \Rightarrow D(\alpha)=c^2D(\beta)$
4. α и β нез. с.в. $\Rightarrow D(\alpha+\beta)=D(\alpha)+D(\beta)$

Док-во

$$2) \qquad \alpha=\beta+c$$

$$E(\alpha)=\sum_{u\in U}\alpha(u)\Pr(u)=\sum_{u\in U}(\beta(u)+c)\Pr(u)=\sum_{u\in U}\beta(u)\Pr(u)+\sum_{u\in U}c\Pr(u)=$$

$$=E(\beta)+c$$

$$D(\alpha)=E(\alpha-E(\alpha))^2=E(\beta+c-E(\beta)-c)^2=D(\beta)$$

$$3) \qquad D(\alpha)=E(c\beta-E(c\beta))^2=E(c\beta-cE(\beta))^2=c^2E(\beta-E(\beta))^2=c^2D(\beta)$$

$$4) \qquad D(\alpha+\beta)=E(\alpha+\beta-E(\alpha+\beta))^2=$$

$$=E(\alpha-E(\alpha)+\beta-E(\beta))^2=E(\alpha-E(\alpha))^2+2E((\alpha-E(\alpha)(\beta-E(\beta)))))+$$

$$+E(\beta-E(\beta))^2=D(\alpha)+D(\beta)+2(E(\alpha\beta-\alpha E(\beta)-\beta E(\alpha)+E(\alpha)E(\beta)))=$$

$$=D(\alpha)+D(\beta)+2(E(\alpha\beta)-E(\alpha)E(\beta))-E(\beta)E(\alpha)+E(\alpha)E(\beta))=D(\alpha)+D(\beta)$$

Схема Бернулли ИСПРАВИТЬ ЭТО. В КОНСПЕКТЕ ЛУЧШЕ. ЗАГЛЯНУТЬ В РОМАНОВСКОГО каком нахуй конспекте блять, нихуя там не лучше

Задача

Стрелок делает 5 выстрелов, какова вероятность того, что он попадет не меньше 4 раз?

p - вероятность попасть в мишень

$$P_5(A) = P_5(4) + P_5(5) = P_5(4) + p^5$$
$$P_5(A) = C_5^4 \cdot p^4(1-p) + p^5$$

hint: мы выбираем, когда стрелок промахнется из всех выстрелов

Опр

$$P_n(m) = C_n^m p^m (1-p)^{n-m} \quad \text{формула Бернулли}$$

n - число попыток $\quad m$ - удачных событий

Случайные числа. Схема Уолкера

Замечание

Зачем все это нужно? Мы хотим провести серию неких эксперементов, для этого мы можем использовать метод статистического моделирование. На компьютере можно имитировать случайные эксперименты. В качестве источника случайности мы можем взять генератор случайных чисел (при каждом обращении генератор дает нам какое-то число). В Романовском написано, что эти величины имеют равномерное распределение, но это зависит только от того, какой генератор взять (Если коротко, то равномерное распределение - это, когда у нас нет "перекосов"в сторону каких-то значений).

Пример

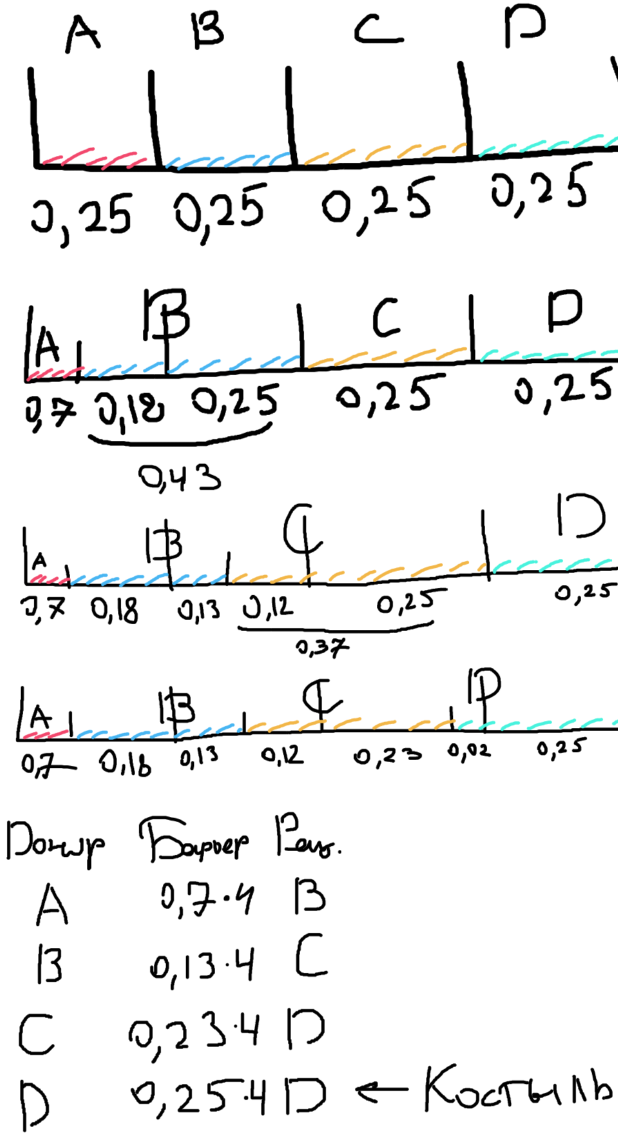
Для вычисления площади $sq(A)$ плоской ограниченной фигуры А можно построить содержащий фигуру прямоугольник R (стороны которого $\|$ осям коорд.) Будем бить случайными точками в этот прямоугольник. Отношение числа точек, попавших в A, к общему числу точек будет хорошей оценкой площади.
При равномерном распределении P(попасть в A) $= sq(A)/sq(R)$

Опр (схема Уолкера)

(По-сути это своеобразный генератор случайных чисел, но с нашим распределением)
Мы хотим создать некую схему, которая сможет нам отвечать, куда мы попали нашей точкой на отрезке $[1, 0]$, какой мы получили исход. Изначально нам даны вероятности этих событий, всего их n. Мы строим таблицу, для каждого исхода запишем $\frac{1}{n}$. То есть, мы изначально предполагаем, что у нас равномерное распределение. Потом мы начинаем это корректировать. Берем событие, которое получило "вероятностной массы" больше, чем остальные. Назовем его донором. Возьмем событие, у которого вероятность ниже, чем мы хотели изначально. Назовем его реципиентом. Отрежем от донора и отдадим реципиенту. (То есть, когда мы будем бить точками в этот отрезанный кусок, то он будет относиться уже к реципиенту, и наш генератор вернет нам другое число). В таблицу нужно еще записать барьеры, которые помогут нам определять, куда попала наша точка. Барьер $=$ остаток донора в $\frac{1}{n}$ отрезка $\cdot$ 4. (то есть, если точка правее, то она попала в реципиента, а иначе в донора) Если у реципиента стало слишком много массы, то он сам станет донором для другого события. (Резать нужно от его исходного куска в $\frac{1}{n}$ - ую). После того, как все исходы получили нужную вероятность, нам остается научиться быстро определять, куда попала наша точка $x \in [0, 1]$. Мы можем быстро понять, в какую $\frac{1}{n}$ -ую попало значение. Умножим x на n и возьмем целую часть. Дальше мы смотрим на барьер, если значение x больше барьера, то выбираем реципиента, иначе донора.

Пример

$$P_a = 0,07 \quad P_b = 0,31 \quad P_c = 0,35 \quad P_d, 27$$



Двоичный поиск и неравенство Крафта ИСПРАВИТЬ ЭТО. В РОМАНОВСКОМ ЛУЧШЕ.

Теорема

Для того чтобы набор из целых чисел от 1 до m мог быть набором длин путей в схеме с m исходами необходимо и достаточно, чтобы:

$$\sum_{i \in 1:m} 2^{-S_i} \leqslant 1, \quad S_i \text{ - числа из набора}$$

Док-во

($\Rightarrow$)

Рассмотрим поисковую схему - двойное дерево T с m листьями, K - вершина, находящаяся на расстоянии t от корня, $a_k = 2^{-t}$, r_0 - корень $\Rightarrow a_{r_0} = 1$

Докажем, что $a_{r_0} \geqslant \sum_{k \in F} a_k$, где F - мн-во листьев

Для каждого нелиста $k \in M \setminus F \Rightarrow a_k \geqslant \sum_{r \in \mathrm{next}(k)} a_r$, где $\mathrm{next}(k)$ - мн-во прямых потомков k

$$\sum_{k \operatorname{Im} M \setminus F} \geqslant \sum a$$

Энтропия. 2 леммы

Опр

Энтропия случайной схемы - мера содержания в этой схеме неопределенности. ρ - вероятностная схема с m исходами, вероятности которых равны $p_1, ..., p_m$

$$H(\{p_1, ..., p_m\}) = \sum_{i=1}^{m} p_i \log_2 \frac{1}{p_i}$$

Свойства

1. Энтропия непрерывно зависит от вероятности при фиксированном m
2. При перестановке в наборе $\{p_1, ..., p_m\}$ энтропия не меняется
3. Нужно ввести шкалу для измерения неопределенности

   Пусть $H(\{\frac{1}{2} : \frac{1}{2}\}) = 1$ - неопределенность схемы с двумя равновероятными событиями
4. При фиксированном m наибольшей неопределенностью обладает схема, в которой все события равновероятны

   $$H(\{\frac{1}{m}, ..., \frac{1}{m}\}) = 1$$
5. $h(m)$ возрастает с ростом m
6. $P(\{p_1, ..., p_m\}) \qquad H(PQ) = H(\rho) p_{n_i} H(Q)$

   $$Q(\{q_1, ..., q_m\})$$

Лемма

$$g(m) = \log_2 m$$

Док-во

$Q_k,\ Q_l$ - две схемы с равновероятными исходами

$$Q_{kl} \qquad (1,1),\ (1,2),\ ...,\ (k,l)$$

$$g(kl) = g(k) + g(l)$$

$$g(m^k) = kg(m)$$

$$g(2^k) = 2g(k)$$

$$S = [\log_2 m^k]$$

$$g(2^s) \leqslant g(m^k) \leqslant g(2^{s+1}) \Rightarrow s \leqslant kg(m) \leqslant s+1$$

$$\Rightarrow 0 \leqslant g(m) - \frac{[k\log_2 m]}{k} = g(m) - \log_2 m + \frac{\{k\log_2 m\}}{k} \leqslant \frac{1}{k}$$

При $k \to \infty \quad g(m) = \log_2 m$

Теорема об энтропии

Теорема

Единственная функция, удовлетворяющая 6-ти св-ам энтропии - это функция $H(\{p_1, ..., p_m\}) = \sum_{i=1}^m p_i \log_2 \frac{1}{p_i}$

Док-во

Обозначим $y_i = 2^{-s_i}$, так что $s_i = \log_2 \frac{1}{y_i}$

В этих обозначениях условие 1 преобразуется к виду:

$$\sum_{i\in 1:m} y_i \leqslant 1, \qquad (@)$$

а целевая функция к виду:

$$T(p,y) = \sum_{i\in 1:m} p_i \log_2 \frac{1}{y_i}$$

Условие 2 перейдет в условие $0 < y_i < 1$

Второе неравенство из этой пары следует только из полученного ограничения на сумму переменных y_i, так что останется только условие положительности.

Целевая функция T состоит из отдельных слагаемых, соответствующих отдельным переменным. Каждое слагаемое убывает с ростом аргумента. Если бы неравенство (@) выполнялось как строгое, то увеличение любой из переменных уменьшило бы значение целевой функции. Поэтому в точке минимума условие 1 выполняется как равенство. Выразим переменную y_m через остальные:

$$y_m = 1 - \sum_{i \in 1:m-1} y_i$$

Подставим её в целевую функцию и приравняем к нулю производные целевой функции $T(p, y)$ по всем оставшимся переменным:

$$\frac{\partial T}{\partial y_i} = -p_i \cdot \log_2 e \cdot \frac{1}{y_i} + p_m \cdot \log_2 e \cdot \frac{1}{y_m}$$

Откуда

$$\frac{p_1}{y_1} = \frac{p_2}{y_2} = \ldots = \frac{p_m}{y_m}$$

Так как обе суммы (p_i и y_i) равны 1, то $y_i = p_i$. Это единственная точка, в которой возможен экстремум функции $T(p, y)$, и следовательно, $\min_y T(p, y) = H(p)$

Операции над строками переменной длины

Опр

А - конечное мн-во, называется алфовитом. Его элементы буквами. Произвольная конечная последовательность букв называется строкой. Кол-во букв в этой последовательности - длина строки

1. Нахождение длины строки

2. Выделение подстроки. Операция выделяет подстроку с заданным числом букв, начиная с заданного места

Опр

Начальная подстрока строки - префикс. Конечная - суффикс. Голова строки (head) - префикс из одной буквы. Остальная часть - хвост (tail)

3. Конкатенация строк (сцепка)

4. Обращение строк (переворот)

5. Сравнение строк по предшествованию, обычно используется лексикографическое сравнение.

   Пусть буквы из A можно сравнивать

   Тогда для строк a и b результат лексикографического сравнения равен

   (a) $a = b$, если a и b - пусты

   (b) $a < b$, если a пустая, а b - нет

   (c) $a > b$, если b пустая, а a - нет

   (d) $a < b$, если $\operatorname{head} a < \operatorname{head} b$

   (e) $a > b$, если $\operatorname{head} a > \operatorname{head} b$

   (f) Результат сравнения $\operatorname{tail} a$ и $\operatorname{tail} b$, если $\operatorname{head} a = \operatorname{head} b$

6. Поиск образца в строке

7. Подстановка (вместо заданного образца нужно вписать другой)

8. Преобразование

   S - строка, $A' = \{a \mid a \in S\}$, $\quad \varphi : A' \Rightarrow V$, $\quad \varphi(B)$ - последовательность элементов из B

9. Фильтрация (все символы делятся на подмн-ва A и B). Если $s_i \in A$, то остается в строке, если $\in B$ - удаляется

10. Слияние

    Пусть на алфавите задан порядок. s_1, s_2 - строки из A. На каждом шаге к результату приписывается $m = \min(\operatorname{head} s_1, \operatorname{head} s_2)$ и удаляется из старого списка

11. Поиск максимального совпадения

Поиск образца в строке (Карпа-Рабина, Бойера-Мура) ИСПРАВИТЬ ЭТО. ДОПОЛНИТЬ.

<u>Опр</u>

Пусть заданы две строки: t (text) и p (pattern). Говорят, что образец входит точно в текст с позиции j, если $t[j : j + m - 1] = p[1 : m]$

Наивный метод: ходим по строке, ищем первый символ, затем второй... Сложность $m \cdot n$

<u>Опр</u>

Пусть задана числовая последовательность $p_1, ..., p_n$, определим скользящую сумму как $s_i = p_i + p_{i+1} + ... + p_{r+i}$, где r - некоторое фиксированное число

<u>Замечание</u>

Нетрудно заметить, что $s_{k+1} = s_k - p_k + p_{k+r}$

ДОПИСАТЬ

<u>Алгоритм</u> (метод Карпа-Раббина)

$o(m + n)$, но по памяти $o(n^2)$ че блять, Романовский, соси хуй

<u>Алгоритм</u> (метод Бойера-Мура)

Сравнение начинается с последнего символа образца, который совмещается с началом. Если совпал $\Rightarrow$ нашли. Если нет, пользуемся эвристиками:

1. Эвристика стоп-символа.

   (a) Если не совпало и текущего символа нет в строке, то следующую проверку можно сдвинуть на длину образа

строка:	.	.	.	.	.	.	$\boxed{П}$	.	.	.	.	.	.	.
шаблон:	К	О	Л	О	К	О	$\boxed{Л}$							
next step:								К	О	Л	О	К	О	Л

   (b) Если такой символ есть, сдвигаемся до последнего вхождения в образце

строка:	.	.	.	.	.	.	$\boxed{К}$	.	.
шаблон:	К	О	Л	О	$\boxed{К}$	О	Л		
next step:			К	О	Л	О	$\boxed{К}$	О	Л

2. Правило хорошего окончания

строка:	...	К	КОЛ	...
шаблон:	КОЛ	О	КОЛ	
next step:			КОЛ	ОЛОКОЛ

Пример для суффиксов и сдвигов: $\varnothing - 1,\ Л - 4,\ ОЛ - 4,\ КОЛ - 4$

<u>Алгоритм*</u> (Кнута-Морриса-Пратта)

Суффиксное дерево
Задача о максимальном совпадении двух строк
Код Шеннона-Фано. Алгоритм Хаффмена. 3 леммы
Сжатие информации по методу Зива-Лемпеля

<u>Опр</u> (Алгоритм)

X - Входная фраза (некая строка, которую мы строим)
Точкой обозначена конкатенация

1. Занести все возможные символы в словарь. (им всем будет присвоен код)
2. Считаем один символ из сообщения и добавим его в X
3. Считаем символ Y, если это символ конца сообщения, то вернем код X (он уже лежит в словаре), иначе:
   (a) Если X.Y уже есть в словаре, то присвоим X = X.Y, перейдем к шагу 3
   (b) Иначе вернем код для X, добавим X.Y в словарь и присвоим входной фразе значение Y X = Y, перейдем к шагу 3

<u>Пример</u>

Пример плохой из-за тупости Григорьевой. На самом деле, если не добавить <u>все</u> возможные символы в словарь, то декодировать строку будет невозможно, либо нам придется передать вместе со строкой еще и словарь, состоящий из односимвольных фраз

$abrakadabra$

$a - 1 \quad b - 2 \quad r - 3 \quad k - 4 \quad d - 5$

$ab - 6 \quad br - 7 \quad ra - 8 \quad ak - 9 \quad ka - 10$

$ad - 11 \quad ab - 12 \quad bra - 13$

Output: $\underset{a}{1}\ \underset{b}{2}\ \underset{r}{3}\ 1\ 4\ 1\ 5\ 1\ 7\ 1$

Опр (Декодирование)

1. Занести все возможные символы в словарь.
2. В X считать первый код сообщения
3. Считать очередной код Y из сообщения, если Y - конец сообщения, то выдать символ, соответствующий коду X, иначе:
   (a) если фразы под кодом X.Y нет в словаре, то вывести фразу, соответствующую коду X, а фразу с кодом X.Y занести в словарь(! но присвоить ей другой код, а именно: кол-во элементов в словаре + 1)
   (b) Иначе присвоить фразе код X.Y и перейти к шагу 3

Метод Барроуза-Уилера

Опр (Алгоритм)

1. Составляется таблица всех циклических сдвигов строки.
2. Производится лексикографическая сортировка строк таблицы.
3. В качестве выходной строки выбирается последний столбец таблицы преобразования и номер строки, совпадающей с исходной.

Пример

Вход	ц. сдвиги	сортировка	Выход
	abacaba	aabacab	
	bacabaa	abaabac	
	acabaab	abacaba	
abacaba	cabaaba	acabaab	bcabaaa, 3
	abaabac	baabaca	
	baabaca	bacabaa	
	aabacab	cabaaba	

BWT("abacaba") = ("bcabaaa 3)

Опр (Алгоритм обратного преобразования)

Пусть нам дали BWT(S) = (A, x)
Тогда выпишем в столбик A, отсортируем, слева допишем A, снова отсортируем, так n раз, где n - длина строки A. После последней сортировки мы получим, что строка с номером x - S

Избыточное кодирование. Коды Хэмминга

Шифрование с открытым ключом

Сортировки (5 методов)

Хеширование

АВЛ-деревья

Опр

АВЛ-дерево - сбалансированное по высоте двоичное дерево поиска, для каждой его вершины высота ее двух поддеревьев различается не более чем на 1

Теорема

АВЛ-дерево с n ключами имеет высоту

$$h = O(\log N)$$

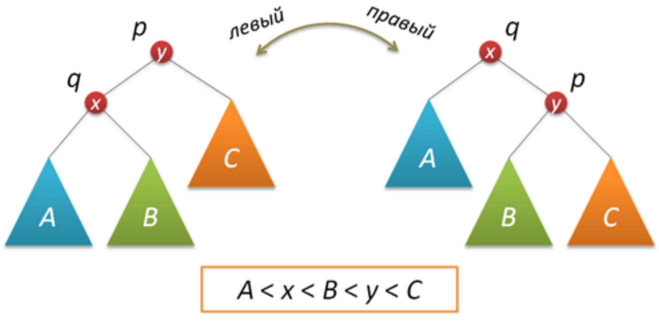
Опр

Баланс вершины - разница между высотами ее поддеревьев

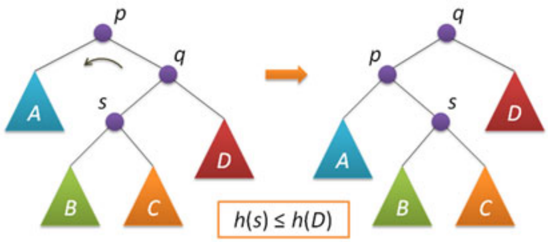
Опр **(Балансировка)**

Балансировка - операция, которая в случае разницы высот левого и правого поддеревьев $|h(L) - h(R)| = 2$, изменяет связи предок-потомок в поддереве данной вершины так, что разница становится $<= 1$, иначе ничего не меняет.
Восстановить баланс можно с помощью поворотов (всего их 4: левый простой, правый простой, левый большой, правый большой)
Простой поворот выполняется при условии $h(s) \leqslant h(D)$

Простой поворот вправо

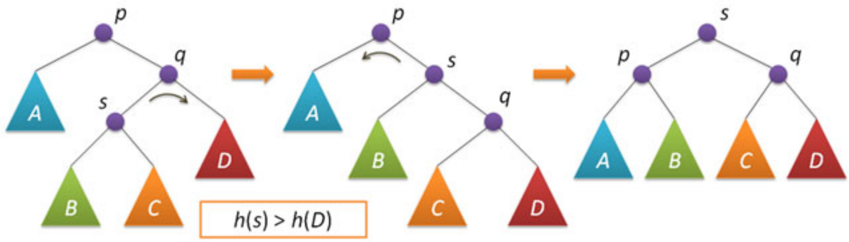


Применение простого поворота



Большой поворот выполняется при условии $h(s) > h(D)$ и сводится к двум простым поворотам

Большой поворот



hint: первым простым поворотом мы отменяем это условие

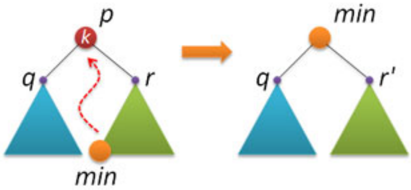
Опр (Добавление вершины)

Добавляем вершину как в бинарном дереве. Спускаемся вниз, как при поиске ключа t. Если мы стоим в вершине a и нам надо идти в поддерево, которого нет, то делаем ключ t листом, а вершину a его корнем.
Дальше поднимаемся вверх и пересчитываем баланс у вершин. Если мы поднялись в вершину i из левого поддерева, то баланс i - ой вершины увеличивается на 1, иначе уменьшается.
Если мы пришли в вершину и ее баланс $= 0$ после пересчета, то это значит, что высота поддерева с корнем в этой вершине не изменилась, можно остановить подъем.
Если баланс вершины после пересчета $= -2$ или 2, то нам необходимо сбалансировать это поддерево.

Добавление работает за $O(\log n)$, т.к. мы рассмотрим не больше, чем $O(h)$ вершин

Опр (Удаление ключа)

1. найдем удаляемый ключ p в дереве (если не нашли, то ничего не делаем)
2. в правом его поддереве найдем наименьший элемент min
3. поменяем местами поменяем местами p и min
4. удалим p
5. сбалансируем все, что выше p



При удалении возможна ситуация, когда вершина p не имеет правого поддерева, тогда по св-ву АЛВ-дерева либо эта вершина имеет слева единственный дочерний узел, либо она является листом. В обоих случаях мы просто удаляем p и возвращаем указатель на левое поддерево.

B-деревья

Биномиальные кучи
Основные определения теории графов

Построение транзитивного замыкания

Обходы графа в ширину и глубину. Топологическая сортировка

Алгоритм поиска контура и построение диаграммы порядка

Теорема о связном подграфе

Теорема

Из связного графа можно выделить подграф - дерево

Опр (Алг. построения)

Аналогичен DFS

Деревья. Теорема о шести эквивалентных определениях дерева

Теорема

1. связный граф без циклов
2. связный граф, в котором дуг на 1 меньше, чем вершин
3. граф без циклов, в котором дуг на 1 меньше, чем вершин
4. минимальный связный граф, т.е граф, который при удалении любого ребра перестает быть связным
5. максимальный граф без циклов
6. граф, в котором между двумя любыми вершинами существует только 1 путь

Задача о кратчайшем остовном дереве. Алгоритм Прима

Алгоритм Краскала

Задача о кратчайшем пути. Алгоритм Дейкстры

Алгоритм Левита

Опр (Алгоритм Левита)

M_0 - вершины, расстоние до которых уже вычислено, (но возможно не окончательно)
M_1 - очередь вершин, расстояние до которых вычисляется
M_2 - вершины, расстояние до которых еще не вычислено
$d[i]$ - расстояние до i-ой вершины

Изначально в M_1 лежит стартовая вершина, в M_2 все остальные.
M_0 - пусто
A d[i] $= +\infty$, кроме $d[start] = 0$

На каждом шаге берем вершину из M_1 (первый элемент в очереди). Пусть V - это выбранная вершина. Переводим эту вершину в M_0. Просматриваем все ребра, выходящие из V. Пусть T - второй конец текущего ребра (то есть не равный V), а L - длины текущего ребра.

Тогда

1. Если T из M_2, то переводим ее в M_1 (в конец очереди).

$$d[T] = d[V] + L$$

2. Если T из M_1, то пытаемся улучшить значение $d[T]$

$$d[T] = \min(d[T], d[V] + L)$$

3. Если T из M_0, и если $d[T]$ можно улучшить, то улучшаем $d[T]$, а вершину возвращаем в M_1

В Романовском используется дополнительная "срочная" очередь M_1'', в которую возвращаются элементы из M_0 и на каждом шаге сначала берут вершину оттуда, а уже потом из основной очереди. Вероятно, такой подход ускорит работу алгоритма.

Задача о кратчайшем дереве путей

Задача

построить остовное дерево на ориентированном графе с корнем в вершине v

Опр (Алгоритм двух китайцев)

Задача о максимальном паросочетании в графе. Алгоритм построения

Опр

Граф $< M, N >$ называется двудольным, если множество его вершин разбито на два множества M_b и M_c и все начала дуг принадлежат M_b, а концы M_c

Опр

набор ребер $J \subset N$ называется паросочетанием, если $\forall j_1, j_2 \in J, \;\; j_1 \neq j_2$ начала и концы этих дуг различны

Опр

максимальное паросочетание - максимальное по числу ребер паросочетание

Опр

Цепью длины k называется некоторый простой путь, содержащий k ребер

Опр

Чередующей цепью (относительно некоторого паросочетания) называется цепь, в которой ребра поочередно принадлежат/не принадлежат паросочетанию.

Опр

Увеличивающей цепью называется чередующаяся цепь, у которой начальная и конечная вершины не принадлежат паросочетанию

Теорема (Бержа)

Паросочетание является максимальным $\Leftrightarrow \nexists$ увеличивающих относительно него цепей

Опр

насыщенная вершина - принадлежащая паросочетанию

Опр (Алгоритм Куна)

Из каждой ненасыщенной вершины графа (из одной доли) будем искать увеличивающую цепь. Для это применим DFS. Мы выбрали ненасыщенную вершину v, рассмотрим все вершины, инцидентные с ней. Назовем текущую вершину to, если она ненасыщенная, то мы нашли увеличивающую цепь, иначе запустим поиск от вершины p $((v, to),\ (to, p))$ пробуем

найти увеличивающую цепь из нее.
Если из вершины p мы нашли увеличивающую цепь, то "прочередуем" ребра. Уберем ребро (p, to) и добавим (v, to)
После того, как все вершины будут просмотрены, текущее паросочетание будет максимальным.

Теорема Кенига

Опр

Вершинное покрытие графа - множество вершин, такое, что любое ребро графа имеет хотя бы одну конечную вершину из этого множества.

Опр

Вершинное покрытие называется наименьшим, если никакое другое вершинное покрытие не имеет меньшего числа вершин.

Теорема (Кёнига)

В любом двудольном графе число ребер в макс. паросочетании равно числу вершин в наименьшем вершинном покрытии

Алгоритм построения контролирующего множества

Задача о назначениях. Венгерский метод
aa

Метод динамического программирования. Задача линейного раскроя

Приближенные методы решения дискретных задач. Жадные алгоритмы

Алгоритмы с гарантированной оценкой точности. Алгоритм Эйлера

Жадные алгоритмы. Задача о системе различных представителей

Приближенные методы решения дискретных задач

Конечные автоматы

Числа Фибоначчи. Производящие функции

Числа Каталана

?Алгоритм Кристофидеса (возможно будет)