Дискретная математика. Теория

Александр Сергеев

1 Булевы функции

Множество - структура, связанная с своими элементами отношением принадлежности или непринадлежности.
(Не является определением)

Элементы множества принадлежат некоторому универсуму U.

Операции над множествами:

1. $A \cup B$ - Объединение
2. $A \cap B$ - Пересечение
3. $A \setminus B$ - Вычитание
4. A^c - Дополненте
5. $A \bigtriangleup B; \quad A \oplus B$ - Исключающее объединение
6. $A \times B$ - Декартово произведение
7. A^k - Декартова степень(вектор)

Свойства декартова произведения:

1. Можно считать, что $A \times A \times A = (A \times A) \times A = A \times (A \times A)$
2. $A^0 = \{()\} = void$

Отношения множеств:

1. $A \subset B$ - включает

2. $A \subseteq B$ - включает или равно(эквивалентно первому в некоторых нотациях)

3. $A = B$ - равенство

2 Отношения множеств. Бинарные отношения

Бинарные отношения - множества пар элементов, которые находятся в отношениях.

Пусть R - бинарное отношение.
$R \subset A \times B$

a и b находятся в отношении R $\Leftrightarrow (a, b) \in R \Leftrightarrow a\,R\,b$

Полное отношение - отношение U^2.

Парадокс Рассела(парадокс брадобрея):
Пусть $A = \{X : X \notin X\}$
Парадоксальность:
$A \in A \Rightarrow A \notin A$
$A \notin A \Rightarrow A \in A$

3 Функции(Отношения)

Функции $\subset$ Отношения
B^A - множество функций из A в B
$f \subset A \times B$

$\forall\ a \in A \quad \exists! b \in B : (a, b) \in f$ - функция(график)

(Комментарий к обозначению B^A: каждому элементу A соответствует один из элементов B. Тогда одна функция задается одной парой из $B^{|A|}$)

Инъекция: $x \neq y \Rightarrow f(x) \neq f(y)$
Сюръекция: $\forall y\ \exists x : f(x) = y$
Биекция = Инъекция $\wedge$ Сюръекция

Свойства отношений:

1. $\forall a : a\,R\,a$ - Рефлексивные
2. $\forall a : \overline{a\,R\,a}$ - Антирефлексивные
3. $a\,R\,b \Rightarrow b\,R\,a$ - Симметричные
4. если $a \neq b$, то $a\,R\,b \Rightarrow \overline{b\,R\,a}$ - Антисимметричные
   $a\,R\,b \wedge b\,R\,a \Rightarrow a = b$
5. $a\,R\,b,\ b\,R\,c \Rightarrow a\,R\,c$ - Транзитивные
6. Рефлексивное $\wedge$ Симметричное $\wedge$ Транзитивное = Отношение эквивалентности
7. Рефлексивное $\wedge$ Антисимметричное $\wedge$ Транзитивное = Частичный порядок(Множество - частично упорядоченное множество/ч.у.м./p.o.set/poset.
   Линейный порядок - $\forall a, b\ a\,R\,b \vee b\,R\,a$)

Теорема
Пусть A - множество, R - отношение эквивалентности на A.
Тогда $\exists$ множество A/R классов эквивалентности: $A/R = \{B | B$ - подмножество A не пересекающееся, $x\,R\,y \Leftrightarrow \exists B \in A/R : x \in B \wedge y \in B\}$.

Определение
$R \subset A \times B$
$S \subset B \times C$
Композиция отношений $T = R \circ S = RS : a\,T\,b \Leftrightarrow \exists c : a\,R\,c \wedge c\,S\,b$
$R^n = R \circ R^{n-1}; R^0 = I$
$R^* = \bigcup\limits_{k=0}^{\infty} R^k$ - рефлексивно-транзитивное замыкание
$R^+ = \bigcup\limits_{k=1}^{\infty} R^k$ - транзитивное замыкание

Теорема
Пусть K - множество всех транзитивных отношений на A, содержащих R.
$\bigcap\limits_{S \in K} S = \operatorname{TrCl} R$.
Тогда $\operatorname{TrCl} R = R^+$
Доказательство
Докажем $\operatorname{TrCl} R \subset R^+$.
$aR^+b, bR^+c \Rightarrow aR^ib, bR^jc \Rightarrow aR^{i+j}c \Rightarrow aR^+c$
$R \subset R^+$.
Тогда $R^+ \in K \Rightarrow \operatorname{TrCl} R \subset R^+$
Докажем $R^+ \subset \operatorname{TrCl} R$.
$\forall S \in K, R^+ \subset S$
По индукции докажем $\forall\, k \geq 1\ R^k \subset S$:

1. $R \subset S$
2. $aR^{k+1}b \Rightarrow \exists\, c:\ aR^kc \wedge cRb \Rightarrow aSc \wedge cSb \Rightarrow a\S b$. Отсюда $R^{k+1} \subset S \Rightarrow \bigcup\limits_{k=1}^{\infty} R^k \subset S \Rightarrow R^+ \subset S \Rightarrow R^+ \subset \bigcap\limits_{S \in k} S$.

Из 1 и 2 $\operatorname{TrCl} R = R^+$, ч.т.д.

Определение
Функциональное отношение $R : R^T R = I$, где I - отношение равенства.
Функциональное отношение $R : a\,R\,b_1 \wedge a\,R\,b_2 \Leftrightarrow b_1 = b_2$.

4 Булевы функции

$\mathbb{B} = 0, 1$
$f : \mathbb{B}^n \to \mathbb{B}$ - n-арная булефа функция.

4.1 Унарные функции

$\mathbb{0}_n$ - тождественный 0: $\forall\, b_1, \ldots, b_n \ \mathbb{0}_n(b_1, \ldots, b_n) = 0$
$\mathbb{1}_n$ - тождественный 1: $\forall\, b_1, \ldots, b_n \ \mathbb{1}_n(b_1, \ldots, b_n) = 1$
id: $\forall\, b \ \ \mathrm{id}(b) = b$
$\neg$ - отрицание: $\forall\, b \ \neg b = 1 - b$

4.2 Бинарные функции

x	y	$\mathbb{0}$	$\wedge$	$\nrightarrow$	x	$\nleftarrow$	y	$\oplus$	$\vee$	$\downarrow$ (nor)	$\equiv$	$\overline{y}$	$\leftarrow$	$\overline{x}$	$\rightarrow$	$\uparrow$	$\mathbb{1}$
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

4.3 Тернарные функции

$< x\ y\ z >$ - медиана(возвращает 1 при 2 и более единицах)
$x\ ?\ y\ :\ z$ - переключатель

4.4 Формулы

Определение
Замыкание множества A - множество функций, которые можно получить с помощью композиции и подстановки функций из A
Определение
Базис или *полная система связок* - система связок, с помощью которой можно задать любую функцию.

Определение
Канонический базис - базис $\{\vee, \wedge, \neg\}$

Определение
Совершенная дизъюнктивная нормальная формула - формула, вида
$f(x_1, x_2, \ldots, x_n) = \ldots \vee \ldots \vee \ldots$, где мы добавляем $\ldots \wedge \neg x_j \wedge \ldots \wedge x_i \wedge \ldots$ в формулу, если $f(\ldots, x_j = 0, \ldots, x_i = 1, \ldots) = 1$
Определение
Множество булевых функций A полное, если любую булеву функцию f

можно выразить через элементы A.
Лемма
A и B - множество булевых функций.
f можно выразить через A.
$\forall\, g \in A$ g можно выразить через B.
Тогда f можно выразить через B.

Следствие
A - базис.
$\forall\, \phi \in A$ ϕ можно выразить через B.
Тогда B - базис.

4.5 Классы Поста

1. F_0 - сохраняющие 0. $\{f \,|\, f(0, \dots, 0) = 0\}$
2. F_1 - сохраняющие 1. $\{f \,|\, f(1, \dots, 1) = 1\}$
3. F_s - самодвойственные функции. $\{f \,|\, \neg f(\neg x_1, \dots, \neg x_n) = f(x_1, \dots, x_n)\}$
4. F_m - монотонные функции. $\{f \,|(\forall\, i\ x_i \le y_i) \Rightarrow f(x_1, \dots, x_n) \le f(y_1, \dots, y_n)\}$
5. F_l - линейная функция. $\{f \,|\, f - \oplus$ от некоторых x_i и $\mathbb{1}\}$

Если все функции A принадлежат к некому классу поста F_*, то все функции, которые можно выразить через A, принадлежат F_*

Определение
Запись функции через $\{\oplus, \wedge, \mathbb{1}\}$ - *Полином Жегалкина.*
Запись xor-ов конъюнкций - *Канонический вид полинома Жегалкина.*

Теорема
У любой булевой функции, кроме тождественного нуля, существует единственный канонический полином Жегалкина.
Доказательство
Он существует, т.к. $\{\oplus, \wedge, \mathbb{1}\}$ - базис.
Слагаемых 2^n от количества аргументов n. Каждое слагаемое может входить или не входить в полином. Тогда канонических полиномов Жегалкина 2^{2^n}, включая пустой. Всего булевых функций тоже 2^{2^n}. Тогда между

полиномами Жигалкина и булевыми функциями биекция, ч.т.д.

Теорема(Поста о полной системе функций)
Пусть $A \nsubseteq F_0, F_1, F_s, F_l, F_m$. Тогда A - базис.
Доказательство
Пусть $f_0(0,\ldots,0) = 1$.

1. $f_0(1,\ldots,1) = 1$
   $f_0(x,\ldots,x) = \mathbb{1}$

2. $f_0(1,\ldots,1) = 0$
   $f_0(x,\ldots,x) = \neg x$

Пусть $f_1(1,\ldots,1) = 0$.

1. $f_1(0,\ldots,0) = 0$
   $f_1(x,\ldots,x) = \mathbb{0}$

2. $f_1(0,\ldots,0) = 0$
   $f_1(x,\ldots,x) = \neg x$

Тогда по случаям:

аа. $\mathbb{1}, \mathbb{0}$
Если f_m не монотонная.
Пусть $x_i \le y_i$.
Тогда $f_m(x_0,\ldots,x_n) = 1$
$f_m(y_0,\ldots,y_n) = 0$
Будем постепенно заменять x_i на y_i, начиная с 1 и до n. В какой-то момент функция сменит значение с 1 на 0. Тогда есть случай, когда
$f_m(y_1, y_2, \ldots, y_{i-1}, x_i, x_{i+1}, \ldots, x_n) = 1$
$f_m(y_1, y_2, \ldots, y_{i-1}, y_i, x_{i+1}, \ldots, x_n) = 0$.
Тогда возьмем функцию $\neg a = f_m(y_1, y_2, \ldots, y_{i-1}, a, x_{i+1}, \ldots, x_n)$, взяв $y_1 \ldots y_{i-1}$ и $x_{i+1} \ldots x_n$ в качестве констант

аб. $\mathbb{1}, \neg, \mathbb{0} = \neg\mathbb{1}$

ба. $\mathbb{0}, \neg, \mathbb{1} = \neg\mathbb{0}$

бб. $\neg$
Пусть $f_s(x_1,\ldots,x_n) = f_s(\neg x_1,\ldots,\neg x_n)$ - не самодвойственная функция.

Тогда найдем нарушение самодвойственности. К примеру, $f_s(0,0,1,1,0) = f_s(1,1,0,0,1)$. Тогда $f_s(\neg x, \neg x, x, x, \neg x) = const$. Тогда мы получили $\mathbb{0}$ или $\mathbb{1}$, а через $\neg$ и второе.

Далее. Возьмем нелинейную функцию $f_l(x, y, \ldots)$ и представим ее в виде полинома Жегалкина. По теореме такой полином единственный, а из нелинейности следует, что хотя бы один член имеет не менее двух аргументов. Выберем минимальный член с не менее 2 аргументами. Выберем первые два члена в нем. Остальные аргументы приравняем к $\mathbb{1}$, а те, что не вошли - приравняем к $\mathbb{0}$. Тогда мы получим один из вариантов

1. $x \wedge y$. Тогда $f(x, y, \ldots)$ - искомый "И".
2. $(x \wedge y) \oplus \mathbb{1}$. Тогда $\neg f(x, y, \ldots)$ - искомый "И".
3. $(x \wedge y) \oplus x = x \wedge (y \oplus \mathbb{1})$. Тогда $f(x, \neg y, \ldots)$ - искомый "И".
4. $(x \wedge y) \oplus y = y \wedge (x \oplus \mathbb{1})$. Тогда $f(\neg x, y, \ldots)$ - искомый "И".
5. $(x \wedge y) \oplus x \oplus y = x \vee y$. Тогда $f(x, y, \ldots)$ - искомый "ИЛИ".

...

Имея $\{\wedge, \neg\}$ или $\{\vee, \neg\}$, можно получить канонический базис.
Отсюда ч.т.д.

Определение
f - *инволюция*, если $f = f^{-1}$

4.6 Схема их функциональных элементов

Теорема о топологической сортировке
В ацикличном ориентированном графе существует нумерация, при которой вершины с меньшими номерами ведут только в вершины с большими номерами
Лемма
В таком графе существует вершина, из которой не выходят ребра.
Доказательство теоремы
Докажем по индукции:

1. Для $n = 1$ верно

2. Для $n > 1$:
   Рассмотрим $n - 1$ вершину, исключая одну такую, из которой не выходят ребра. Пронумеруем их от 1 до $n-1$ в соответствии с утверждением. Добавим удаленную вершину, присвоив ей номер n. Из нее не выходят вершин и ее номер наибольший. Тогда утверждение верно, ч.т.д.

Определение
Выберем базис связок F
Схема из функциональных элементов - это ацикличный ориентированный граф с кратными ребрами, в котором каждые входящие в вершину ребра пронумерованы.

СФЭ позволяют строить схемы функций.
Изначально у нас есть вершины $x_1, \ldots, x_n$ - аргументы нашей функции. Из аргументов идут ребра к вершинам, символизирующим функции из нашего базиса F(порядок входа ребер важен, количество входящих ребер соответствует количеству аргументов функции). Далее из любых вершин еще могут выходить ребра. Результат нашей функции символизируется одной из перечисленных вершин.

Теорема
Любую функцию можно задать СФЭ.
Доказательство
СФЭ - дерево разбора, направленное снизу вверх с объединением листьев в n вершин.

Пусть $\mathrm{size}_A\, f$ - минимальное количество внутренних элементов СФЭ в схеме для f над базисом A.
Теорема
Пусть A, B - базисы
Тогда $\exists C\ \forall f\ \ \mathrm{size}_A\, f \le C \cdot \mathrm{size}_B\, f$
Доказательство
Построим СФЭ функции f в базисе B. Выразим все функции из B через A. Пусть C - максимальное количество элементов, которое мы использовали на одну функцию из B. Тогда одна вершина в исходной СФЭ заменилась не более чем на C. Тогда мы получили СФЭ из не менее $C \cdot \mathrm{size}_B\, f$. Тогда $\mathrm{size}_A\, f \le C \cdot \mathrm{size}_B\, f$, ч.т.д.

Определение
Глубина схемы - максимальная длина пути в СФЭ.

Пусть $\operatorname{depth}_A f$ - минимальная глубина СФЭ в схеме для f над базисом A.
Теорема
Пусть A, B - базисы
Тогда $\exists C\ \forall f\ \ \operatorname{depth}_A f \le C \cdot \operatorname{depth}_B f$

4.7 Минутка АрхЭВМ

4.7.1 Линейный сумматор

Полусумматор - $f(x_1, x_2) = (carry = x_1 \wedge x_2, sum = x_1 \oplus x_2)$
Полный сумматор - $f(x_1, x_2, x_3) = (carry = \langle x_1 x_2 x_3 \rangle, sum = x_1 \oplus x_2 \oplus x_3)$
Линейный сумматор двоичных чисел можно построить каскадом сумматоров.
Размер - $O(n)$, глубина - $\Omega(n)$

4.7.2 Двоичный каскадный сумматор

Рассмотрим $f_i : c_{i+1} = f_i(c_i)$

x	y	f
0	0	$\mathbb{0}$ = k(kill)
0	1	id = p(propagate)
1	0	id = p(propagate)
1	1	$\mathbb{1}$ = g(generate)

Рассмотри $f_1(f_2(x))$(столбец - f_1, строка f_2):

f	k	g	p
k	k	g	k
g	k	g	g
p	k	g	p

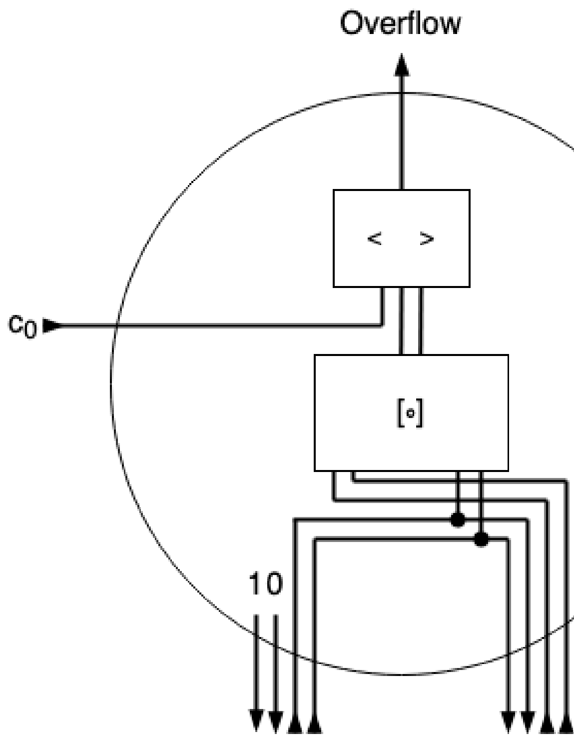
Отсюда

$$\begin{aligned} \dots kpp\dots p &= k \\ \dots gpp\dots p &= g \\ ppp\dots\dots p &= p \end{aligned}$$

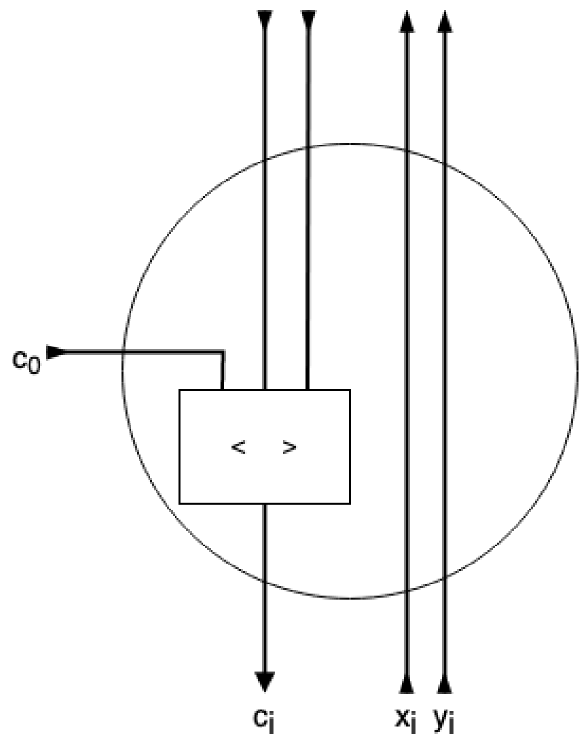
Пусть количество аргументов $n = 2^m$

Построим двоичное полное дерево, которое отвечает за определенное количество битов.

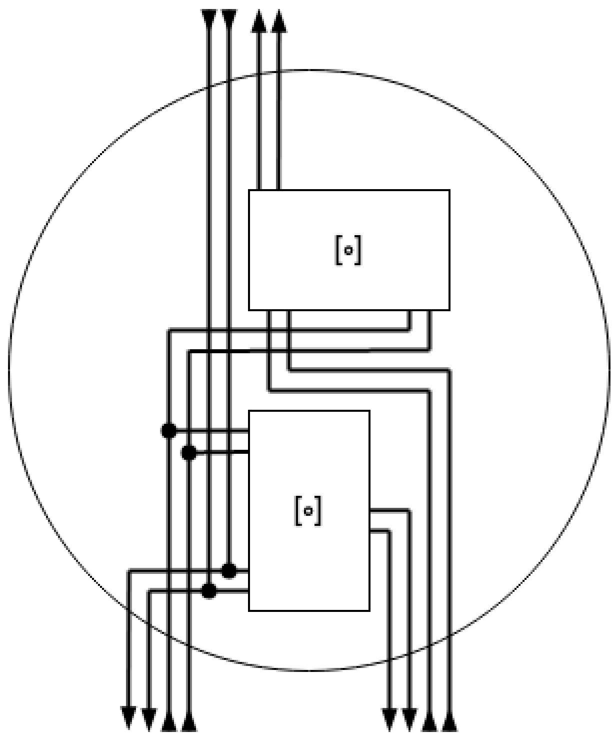
Корень:



Лист:



Узел:



Здесь $[\circ]$ является блоком композиции, работающим по таблице выше. Нетрудно заметить, что в итоге на i-ый лист поступит композиция функций вида $f_{i-1} \circ f_{i-2} \circ \ldots \circ f_0$

Тогда на выходах листьев мы получим $c_0 \ldots c_{n-1}$. В корне же мы получим параметр Overflow, указывающий, произошло ли переполнение

Сложив каждый c_i с x_i и y_i, мы получим искомую сумму

Количество вершин $2n - 1 = O(n)$

Глубина $O(\log n)$

4.7.3 Вычитание

Чтобы выполнить вычитание x и y, нужно сделать сложение $x + \overline{y} + 1$. Сложим x и $\overline{y}$, подав 1 на c_0

4.7.4 Умножение

Будем выполнять умножение первого числа на каждый разряд второго. В сумме блок, выполняющий это действие, будет иметь размер $O(n^2)$, и глубину $O(1)$

Теперь научимся складывать имеющиеся n чисел

Сумматор 3 в 2 - устройство, сопоставляющее числам x, y, z числа u и v так, что $x+y+z=v+u$ и имеющее размер $O(n)$ и глубину $O(1)$. Таким сумматором является полный сумматор

Дерево Уоллеса

Циклически будем подавать тройки значений на сумматоры 3 в 2, получив в результате два числа, которые сложим

Размер - $O(n^2)$

Глубина - $O(\log n)$

В итоге мы получаем схему умножения x на y:

Первый блок генерирует n чисел вида $x \wedge y_i << i$, которые суммируем деревом Уоллеса

Суммарный размер $O(n^2)$, глубина $O(\log n)$

4.8 Оценка размера представления функции

Теорема

Для любой булевой функции от n аргументов достаточно $O(\frac{2^n}{n})$

или

Для любого базиса $\exists C\ \forall 0 < \varepsilon < 1\ \exists n_0$: если $n > n_0$ и используется $< C \cdot \frac{2^n}{n}$ функциональных элементов, то можно реализовать $\leq \varepsilon \cdot 2^{2^n}$ функций

Доказательство

Выберем базис $B = \{\downarrow\}$

Рассмотрим линейную программу и рассмотрим сколько программ мы можем сделать

$$\begin{array}{ll} x_{n+1} = x_{i_1} \downarrow xj_1 & n^2\text{способов} \\ x_{n+2} = x_{i_2} \downarrow xj_2 & (n+1)^2\text{способов} \\ \vdots & \vdots \\ x_{n+k} = x_{i_k} \downarrow xj_k & (n+k)^2\text{способов} \end{array}$$

Тогда количество функций $\leq$ число программ $\prod_{i=1}^{k}(n+i-1)^2 \leq (n+k)^{2k} =$
$2^{2k\log_2(n+k)} \leq 2^{3C\cdot 2^n} \overset{*}{=} (2^{2^n})^{3C}$
(*) Пусть $k = C\dfrac{2^n}{n}$
$2k\log_2(n+k) = \dfrac{2C\cdot 2^n}{n}\log_2(n+\dfrac{C\cdot 2^n}{n}) \leq \dfrac{2C\cdot 2^n}{n}log_2(C\cdot 2^n) = \dfrac{2C\cdot 2^n(\log_2 C + n)}{n} =$
$\dfrac{d\cdot 2^n}{n} + 2c\cdot 2^n \leq 3C\cdot 2^n$
(т.к. $\dfrac{d}{n} \to 0$)
Доля функций, которые можно реализовать за менее $O(\dfrac{2^n}{n})$, стремится к 0
Теперь докажем, что $C\cdot\dfrac{2^n}{n}$ достаточно для большинства функций
Рассмотрим $s-k$-разложение Лупанова
Пусть у нас есть функция от n аргументов
Назовем аргументы: $f(x_1,\ldots,x_k,y_1,\ldots,y_{n-k})$
Построим прямоугольную таблицу истинности:

	$0\ldots00$	$0\ldots01$	$\cdots$	$1\ldots11$
$0\ldots00$				
$0\ldots01$				
$\vdots$				
$1\ldots11$				

Нарежем таблицу на горизонтальные полосы ширины s. Всего их $p = \left\lceil\dfrac{2^n}{s}\right\rceil$
Теперь обнулим все полосы, кроме i-ой. Всего мы можем получить $p\cdot 2^s$ функций(т.к. не более 2^s вариантов столца высоты s). Назовем такие функции $g_{i,mask}$, где $mask$ - значения
$f(x_1,\ldots,x_k,y_1,\ldots,y_{n-k}) = \bigvee_{i=1}^{p} g_{i,mask[y_1,\ldots,y_{n-k}]}(x_1,\ldots,x_k)$
Возьмем демультиплексор, адресом для которого будут выступать $x_1\ldots x_k$. На вход ему подадим единицу. Каждый выход будет соответствовать одной строчке в нашей таблице. Тогда 1 будет только на том выходе, которому соответствуют наши $x_1\ldots x_k$. На такой демультиплексор уйдет 2^k

элементов. Далее для каждой полосы соберем все возможные функции g, взяв or от всех строк данной полосы, на которых g равна 1. На одну такую функцию уйдет не более s операций or. Всего таких функций 2^s на одну полосу, а полос $\frac{2^k}{s}$. Итого на все это уйдет $s2^s\frac{2^k}{s} = 2^{s+k}$. Далее для каждого набора $y_1 \dots y_{n-k}$ объединим все g в один блок через or, которые соответствуют данному набору. Всего наборов 2^{n-k}, а g - $\frac{2^k}{s}$. Отсюда на этот фрагмент уйдет $\frac{2^n}{s}$ элементов. Потом с помощью мультиплексора, где адресом будет $y_1 \dots y_{n-k}$, выберем нужный нам набор. На это уйдет 2^{n-k} элементов.
Обобщая:

1. Подадим 1 на выход демультиплексора, соответствующей строке $x_1 \dots x_k$
2. Для каждой полосы i создадим функцию $g_{i,m}$ для всех возможных уникальных m
3. Для каждого набора $y_1 \dots y_{n-k}$ объединим все функции $g_{*,m} : m = mask[y_1 \dots y_{n-k}]$
4. Через мультиплексор выберем среди всех объединений то, что соответствует нашим $y_1 \dots y_{n-k}$

Итого мы можем собрать схему за $2^k + 2^{s+k} + \frac{2^n}{s} + 2^{n-k}$

Выберем $\begin{cases} k = 2\log_2 n \\ s = n - 3\log_2 n \end{cases}$

Тогда $\begin{cases} 2^k = n^2 \\ 2^{s+k} = \frac{2^n}{n} \\ \frac{2^n}{s} \geq \frac{2 \cdot 2^n}{n} \\ 2^{n-k} = \frac{2^n}{n^2} \end{cases}$

Тогда суммарная асимптотика $O(\frac{2^n}{n})$, ч.т.д.

Т.о. за такую асимптотику точно возможно реализовать функцию.
Объединяя обе теоремы, получаем, что для большинства функций size $f = \Theta(\frac{2^n}{n})$

5 Представление информации

5.1 Код. Код Хаффмана. Неравенство Крафта-Макмилана

Определение
Алфавит - произвольное конечное непустое множество (обозначаются Σ)
Буква или *Символ* - элементы этого множества (обозначаются $a, b, c, \dots$)
Множество слов над алфавитом Σ или *цепочка* или *строка* - $\sum\limits_{i=0}^{\infty} \Sigma^i$ (обозначают u, v, w, x, y, z или $\alpha, \beta, \gamma, \dots$)
Конкатенация $\cdot : \Sigma^* \times \Sigma^* \to \Sigma^*$ - получение строки путем приписывания второго операнда к концу первого
Свойства конкатенации

1. $(\alpha\beta)\gamma = \alpha(\beta\gamma) = \alpha\beta\gamma$
2. $\exists \varepsilon \in \Sigma^0 : \ \alpha\varepsilon = \varepsilon\alpha = \alpha$ - нейтральный элемент

Множество $(\Sigma, \cdot)$ - *моноид* (на самом деле даже *свободный* моноид над Σ)
c - *код* над Σ, если существует $c : U \to \Sigma^*$
Код *однозначно декодируемый*, если c - биекция
Если $U = \Pi^*$:
Тогда $c : \Pi^* \to \Sigma^*$
Код разделяемый, если c - гомоморфизм(т.е. $c(\alpha \cdot \beta) = c(\alpha) \cdot c(\beta)$)

Если код - однозначно декодируемый бинарный код постоянной длины: $|c(a \in \Pi)| = \lceil \log_2 |\Pi| \rceil$

Теперь попробуем сделать код непостоянной длины
Пусть $f : \Pi \to \mathbb{N}$ - частота появления каждой буквы алфавита Π в неком тексте

Минимизируем $\sum\limits_{a \in \Pi} f(a)|c(a)| = \sum\limits_{a \in \Pi} f(a)l(a)$ для данного текста, где $l(a) = |c(a)|$

Код Хаффмана - оптимальный префиксный бинарный код

Определение

Код называется *префиксным*, если $a \neq b \Rightarrow c(a)$ - не префикс $c(b)$

Лемма

Префиксный код однозначно декодируемый

Доказательство

Пусть это не так. Тогда $\exists\, \alpha\beta \in \Pi^* : \; c(\alpha) = c(\beta)$

Пусть строчки различаются с i-ого символа. $c(\alpha_i)$ - префикс $c(\beta_i)$ или наоборот, что противоречит условию, ч.т.д.

Неравенство Крафта-Макмилана

Однозначно декодируемый разделяемый бинарный код с длинами слов $l_1, l_2, \ldots, l_n$ существует $\Leftrightarrow \sum\limits_{i=1}^{n} 2^{-l_i} \leq 1$

Доказательство $\Leftrightarrow$

Следует из Леммы

Доказательство $\Rightarrow$

Доказать: если код однозначно декодируемый $\Rightarrow \sum\limits_{i} 2^{-l_i} \leq 1$

Выберем $\Pi = \{a, b\}$

Выберем кодовые слова $\alpha_1, \alpha_2, \ldots, \alpha_n$, являющиеся однозначно декодируемые

Рассмотрим полукольцо над словами

Сложим слова $S = \alpha_1 + \alpha_2 + \ldots + \alpha_n$ и возведем в k-ую степень

$S^k = (\alpha_1 + \alpha_2 + \ldots + \alpha_n)^k$

Мы получили сумму n^k различных произведений (конкатенаций). Никакие два члена не равны из однозначности декодируемости

Пусть $a = b = \frac{1}{2}$

Тогда $\alpha_1 + \alpha_2 + \ldots + \alpha_n = \sum\limits_{i=1}^{n} (\frac{1}{2})^{l_i}$, где l_i - длина α_i

Длины всех всех произведений от k (как минимум) до $Lk : L = \max\limits_{i} l_i$

Сгруппируем все произведения длины k. Их количество $\leq 2^k$, а каждый член произведения равен $\frac{1}{2^k}$. Тогда сумма каждой группы не больше 1

Отсюда $S^k \le Lk$

Но $S = \sum\limits_{i=1}^{n} 2^{-l_i}$

Тогда $\forall\, k\ (\sum\limits_{i=1}^{n} 2^{-l_i})^k \le Lk$

Если $S > 1$, то с некоторого места $S^k > Lk$

Отсюда $S \le 1 \Leftrightarrow \sum\limits_{i=1}^{n} 2^{-l_i} \le 1$, ч.т.д.

Лемма

$\sum\limits_{i=1}^{n} 2^{-l_i} \le 1 \Rightarrow$ существует оптимальный префиксный код с длинами $l_1, \ldots, l_n$

Доказательство

Упорядочим $l_i : l_1 \le l_2 \le \ldots \le l_n$

Возьмем отрезок длины 1 и последовательно отложим слева отрезки длин 2^{-l_i}

Утверждается, что если сумма длин отрезков больше $\frac{1}{2}$, то точка $\frac{1}{2}$ на отрезке - граница какого-то отрезка

Доказем это:

Выберем отрезок j такой, что

$2^{-l_1} + 2^{-l_2} + \ldots + 2^{-l_j} \le \frac{1}{2}$

$2^{-l_1} + 2^{-l_2} + \ldots + 2^{-l_j} + 2^{-l_{j+1}} > \frac{1}{2}$

Домножим оба выражения на $2^{l_{i+1}}$

Отсюда

$2^{l_{j+1}-l_1} + 2^{l_{j+1}-l_2} + \ldots + 2^{l_{j+1}-l_j} \le 2^{l_{j+1}-1}$

$2^{l_{j+1}-l_1} + 2^{l_{j+1}-l_2} + \ldots + 2^{l_{j+1}-l_j} + 1 > 2^{l_{j+1}-1}$

В обоих выражениях обе части целые. Из этого следует, что в первом выражении равенство, отсюда утверждение доказано

Теперь разделим отрезок пополам. Пусть коды в левой половине отрезка начинаются с 0, а в правой - с единицы

Тогда задача сводится к построению префиксного кода длин $l_i - 1$ для левой и правой частей отдельно

Т.о. мы построили префиксный код, т.е. лемма доказана

Лемма 1

Существует оптимальное дерево, в котором $x, y : f_x, f_y$ минимальные —

братья на максимальной глубине
Доказательство
Рассмотрим вершину a на максимальной глубине.

1. У нее есть брат b: если бы его не было, дерево было бы неоптимальным

2. если a, b имеют не минимальные f_a, f_b: поменяем местами a, b местами с x, y, имеющими минимальные f_x, f_y. Тогда сумма $\sum\limits_{a \in \Pi} f(a)l(a)$ не увеличится (к примеру, по перестановочному неравенству), а значит мы получим оптимальное дерево. Отсюда такое дерево существует

Алгоритм Хаффмана
$\Pi = \{a_1, a_2, \ldots, a_n\}$
Построим *бор* - дерево префиксного кода (в вершине хранится буква. Лист соответствует строке, полученной конкатенацией всех символов от корня до данного листа)

1. Для $n = 2$ оптимальный бор - дерево из корня и двух листов

2. Для $n > 2$: Выберем x, y - два символа с минимальными f_x и f_y
Заменим их на символ z
$f_z = f_x + f_y$
Мы получили алфавит Π'. Решим для него задачу построения минимального кода с суммой Φ'. Затем сделаем замену $c(x) = c(z) \cdot 0, c(y) = c(z) \cdot 1$
После замены мы получили сумму
$\Phi' = \sum\limits_{a \in \Pi \setminus \{x,y\}} f(a)l(a) + f_z l_z = \sum\limits_{a \in \Pi \setminus \{x,y\}} f(a)l(a) + (f_x + f_y)(l_x - 1) = \sum\limits_{a \in \Pi \setminus \{x,y\}} f(a)l(a) + f_x l_x + f_y l_y - (f_x + f_y) = \Phi - (f_x + f_y)$, где Φ - сумма для n элементов
Отсюда $\Phi = \Phi' + f_x + f_y$
Т.к. Φ' - минимальное по индукционному переходу(сумма для кода из $n - 1$ элементов), а f_x, f_y - по условию, то Φ - минимальное

5.2 Арифметическое кодирование

Кодирование

Возьмем слово S. Посчитаем количество вхождений каждого символа в нем

Возьмем определенный порядок $a, b, c, \ldots$ символов в алфавите

Возьмем отрезок длины 1 и поделим его в отношении количества вхождений $f_a, f_b, f_c, \ldots$

Возьмем подотрезок, соответствующий S_1, и проделаем с ним ту же операцию

В данном подотрезке возьмем подотрезок, соответствующий S_2, и сделаем то же самое

Проделаем это последовательно для всех символов S_i

В последнем отрезке выберем точку $\frac{p}{2^q}$ с минимальным q

Тогда кодом данного слова будет бинарный код длины 2^q со значением p

Декодирование

Зная порядок $a, b, c, \ldots$ и отношение $C_a : C_b : C_c : \ldots$, будет разбивать отрезок длины 1 в отношении $f_a : f_b : f_c : \ldots$ и определять, какому отрезку принадлежит $\frac{p}{2^q}$

Проделав такую операцию $|C_a + C_b + C_c + \ldots|$ раз, получим значение слова S

Оценка

Оценим длину результирующего отрезка

Пусть $C(S)$ - код

$\operatorname{len} C(S) = q$

Заметим, что $q = \lceil -\log_2 R - L \rceil$ точно подойдет

Тогда $q \leq \lceil -\log_2 R - L \rceil$

$R - L = 1 \cdot \frac{f_{S_1}}{m} \cdot \frac{f_{S_2}}{m} \cdot \ldots = \frac{\prod_{i=1}^n f_i^{f_i}}{m^m}$, где $m = |S|$

$R - L = \frac{\prod_{i=1}^n f_i^{f_i}}{m^m} = \sum_{i=1}^n f_i \log_2 f_i - m \log_2 m = \sum_{i=1}^n f_i \log_2 f_- \sum_{i=1}^n f_i log_2 m = m \sum_{i=1}^n \frac{f_i}{m} \log_2 \frac{f_i}{m}$

$p_i = \frac{f_i}{m}$ - вероятность вхождения

$R - L = m \sum_{i=1}^{n} p_i \log_2 p_i$

$q \leq m \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i} = m \cdot H(p_1, \ldots, p_n)$

5.3 Алгоритмы семейства LZ

Разделим токены на 2 типа: символы и ссылки (d_i, f_i) - повтори повтори последние d_i повторяя их, выпиши f_i символов
TODO

5.3.1 LZW

TODO

5.3.2 Move to front

TODO

5.3.3 Алгоритм Барроуза-Уилера

TODO
При применении этого алгоритма мы из строки с большим количеством повторяющихся подстрок получим строку с большим количеством подряд идущих символов, что позволяет кодировать ее алгоритмами MoF, LZ, LZW

5.4 Избыточное кодирование

Будем рассматривать только разделяемые коды постоянной длины и решать задачу обнаружения и исправления ошибок
Из ошибок будем рассматривать только ошибки замены символа другим
Для этого будем использовать *контрольные суммы*

5.4.1 Расстояние Хемминга

$H(s, t) = \sum_{i=1}^{n} |s_i - t_i|$ - Расстояние Хемминга
Докажем, что расстояние Хемминга - метрика:

Для этого докажем неравенство треугольника:
$(H(x, z) \le H(x, y) + H(y, z)$
Заметим, что для каждого символа $H(x_i, z_i) \le H(x_i, y_i) + H(y_i, z_i)$
Остальные аксиомы очевидны
Тогда и для строк это выполняется, ч.т.д.

Пусть $|\Sigma| = n$
$c_1, c_2, \dots, c_n$ - слова
$|c_i| = m$

Шар $S(x, r) = \{y : H(x, y) \le r$ Утверждается, что код c обнаруживает d ошибок, если
$\forall\, i \ne j\ H(c_i, c_j) > d$
(Все возможные коды в данном кодировании с удалены друг от друга более чем на d)
Действительно, если произошло d ошибок, то код с ошибками будет удален от нашего на d. Тогда мы из одного реального кода в c не можем попасть в другой
Тогда в случае получения d ошибок мы гарантированно не получим "нормальный"код
Определение
Код c *исправляет* d ошибок, если
$\forall\, i \ne j\ H(c_i, c_j) > 2d$

В таком случае код с ошибками всегда будет удален от оригинала меньше, чем от других кодов. Тогда мы можем однозначно определить оригинал
Эквивалентное утверждение
Код c исправляет d ошибок, если
$\forall\, i \ne j\ S(c_i, d) \cap S(c_j, d) = \varnothing$

Пусть $|S(x, d)| = \sum\limits_{i=1}^{n} \binom{N}{K}$ - объем шара
$\sum\limits_{i=1}^{n} |S(c_i, d)| \le 2^m$
Заметим, что радиусы шаров одинаковые
Отсюда $nS(x, d) \le 2^m$, где x - любой код

(Граница Хемминга) Граница Хемминга - необходимое условие существования кода, исправляющего ошибки
TODO Граница Гильберта
Граница Гильберта - достаточное условие существования кода, исправляющего ошибки

Теорема
Для любого d существует код, обнаруживающий исправляющий d ошибок
Для исправления можно передавать каждый бит $2d + 1$ раз
Для определения - $d + 1$ раз

Рассмотрим код, исправляющий 1 ошибку - *Код Хемминга*:
Будем кодировать исходную строку длины k
Пусть в нашем коде биты с номерами(считая от 1), равными степени 2 - *контрольные биты*, остальные - *информационные*
Заметим, что контрольных битов $\approx \log_2 k$
Отсюда длина конечного кода $m \approx k + \log k$
Такой код будет очень близок к нижней границе - границе Хемминга
В информационные биты последовательно занесем нашу строку
В коде b подберем контрольные биты так, что $\forall\, j \quad \bigoplus\limits_{\substack{i=1 \\ i\&(1<<j)\neq 0}}^{m} b_i = 0$
Заметим, что контрольные биты друг на друга не влияют
Номер поврежденного бита $e = \sum\limits_{i\text{-ый к. бит повр}} i$

Теорема
В коде Хемминга $H(c_i, c_j) \geq 3$
Доказательство
TODO

6 Комбинаторика

Определение
Пусть у нас есть алфавит Σ и отношение линейного порядка $\leq$ на нем
Лексикографическим порядком над множеством слов Σ^* будет являться отношение $\leq$ такое что

1. Если a - префикс b, то $a \le b$, где $a, b \in \Sigma^*$
2. Если $\exists\, j \le |a|, |b| : i < j \Rightarrow a_i = b_i, a_j < b_j$, то $a \le b$, где $a, b \in \Sigma^*$

Теорема
Лексикографический порядок - линейный порядок над Σ^*

Определение
Код Грея - такой порядок над $\mathbb{B}^n$, что любые два соседних элемента различаются в 1 разряде
Циклический код Грея - код Грея, где первый и последний элемент различаются в 1 разряде

Зеркальный код Грея: Пусть g_i - i-ый элемент в зеркальном коде Грея длины $n-1$. $|g_i| = 2^{n-1} = a$
Построим код Грея G_i длины n:

$$\begin{array}{ccc} G_0 & = & 0g_0 \\ G_1 & = & 0g_1 \\ & \vdots & \\ G_{a-1} & = & 0g_{a-1} \\ G_a & = & 1g_{a-1} \\ G_{a+1} & = & 1g_{a-2} \\ & \vdots & \\ G_{2a-1} & = & 1g_0 \end{array}$$

Другими словами, выпишем g, приписав к каждому элементу слева 0, а затем выпишем g в обратном порядке, приписав 1
Зеркальный код Грея - циклический код Грея

Определение
Перестановка множества A - последовательность элементов из A, где каждый элемент A встречается ровно 1 раз
Пусть $n = |A|$
Тогда количество перестановок $P_n = n \cdot (n-1) \cdot (n-2) \cdot \ldots \cdot 2 \cdot 1 = n! = nP_{n-1}$

Определение
Инверсия - ситуация, когда больший элемент в векторе стоит до меньшего

Определение
Размещение - последовательность элементов из B, где каждый элемент B встречается не более одного раза раз
Количество размещений длины k $A_n^k = n \cdot (n-1) \cdot \ldots \cdot (n-k+1) = \dfrac{n!}{(n-k)!} = n^{\underline{k}}$

Определение
a в k-ой убывающей степени - $a^{\underline{k}} = a \cdot (a-1) \cdot \ldots \cdot (a-k+1)$ - k убывающих множителей
a в k-ой возрастающей степени - $a^{\overline{k}} = a \cdot (a+1) \cdot \ldots \cdot (a+k-1)$ - k возрастающих множителей
$a! = 1^{\overline{n}} = n^{\underline{n}}$

Определение
Сочетание размера k - подмножество элементов размера k
Тогда количество сочетаний $C_n^k = \dbinom{n}{k} = \dfrac{A_n^k}{P_k} = \dfrac{n!}{(n-k)!k!}$
Каноническое представление сочетания - возрастающая перестановка сочетания
Свойства

1. $\dbinom{n}{k} = \dbinom{n}{n-k}$
2. $\dbinom{n}{k} = \dbinom{n-1}{k} + \dbinom{n-1}{k-1}$ - треугольник Паскаля
3. $(a+b)^n = \displaystyle\sum_{i=0}^{n} \binom{n}{i} a^i b^{n-i}$

Теорема(формула включений-исключений)
$|A_1 \cup A_2 \cup \ldots \cup A_n| = \displaystyle\sum_{\varnothing \neq I \subset \{1,\ldots,n\}} (-1)^{|I|+1} |\bigcap_{i \in I} A_i|$
Доказательство
$|A_1 \cup A_2 \cup \ldots \cup A_n| =$
$|(A_1 \cup A_2 \cup \ldots \cup A_{n-1}) \cup A_n| =$
$|A_1 \cup A_2 \cup \ldots \cup A_{n-1}| + |A_n| - |(A_1 \cup A_2 \cup \ldots \cup A_{n-1}) \cap A_n| =$
$\displaystyle\sum_{\substack{I \subset \{1,\ldots,n-1\} \\ I \neq \varnothing}} (-1)^{|I|+1} |\bigcap_{i \in I} A_i| + |A_i| - |A_1 \cap A_n \cup A_2 \cap A_n \cup \ldots \cup A_{n-1} \cap A_n| =$

$$\sum_{\substack{I\subset\{1,\dots,n-1\}\\ I\neq\varnothing}}(-1)^{|I|+1}|\bigcap_{i\in I}A_i|+|A_i|-\sum_{\substack{I\subset\{1,\dots,n-1\}\\ I\neq\varnothing}}(-1)^{|I|+1}|\bigcap_{i\in I}A_i\cap A_n|=$$
$$\sum_{\substack{I\subset\{1,\dots,n\}\\ n\notin I, I\neq\varnothing}}(-1)^{|I|+1}|\bigcap_{i\in I}A_i|+\sum_{\substack{I\subset\{1,\dots,n\}\\ I=\{n\}}}(-1)^{|I|+1}|\bigcap_{i\in I}A_i|+\sum_{\substack{I\subset\{1,\dots,n\}\\ n\in I, I\neq\{n\}}}(-1)^{|I|+1}|\bigcap_{i\in I}A_i|=$$
$$\sum_{\substack{I\subset\{1,\dots,n\}\\ I\neq\varnothing}}(-1)^{|I|+1}|\bigcap_{i\in I}A_i|, \text{ ч.т.д.}$$

6.1 Алгоритмы генерации

TODO

6.2 Числа Каталана

6.2.1 Правильные скобочные последовательности

Определение 1 (подход на языке порождений)

1. Пустая строка - правильная скобочная последовательность (далее п.с.к.)
2. (п.с.к.) – п.с.к.
3. п.с.к. + п.с.к. – п.с.к.

Определение 2 (подход распознавания)
Пусть баланс - разность между открывающими и закрывающими скобками
Тогда п.с.к. - это с.к., суммарный баланс которой равен 0, а на всех префиксах неотрицательный

П.с.к. можно сопоставить с путем Дика(построим график баланса от длины префикса)
Определение
Числа Каталана C_n - количество п.с.к. длины n
Заметим, что C_n - это количество "(п.с.к. длины n-1)"+ все "п.с.к. длины i + п.с.к. n-i"
Проблема: среди "п.с.к. длины i + п.с.к. n-i"могут быть те, что разбиваются на две неоднозначно(к примеру, "()()()"). Тогда требуется не

считать их несколько раз
Чтобы правильно посчитать "п.с.к. длины i + п.с.к. n-i потребуем, чтобы первая п.с.к. не разбивалась на две
Тогда "п.с.к. длины i + п.с.к. n-i- "(п.с.к. длины i-1) + п.с.к. n-i"
Отсюда $C_n = C_{n-1} + \sum_{i=1}^{n-1} C_{i-1}C_{n-i} = \sum_{i=1}^{n} C_{i-1}C_{n-i}$
Первые числа Каталана: $C_0 = 1, C_1 = 1, C_2 = 2, C_3 = 5, C_4 = 14, C_5 = 42, \ldots$
Также научимся считать $A_{m,b}$ - количество п.с.к., являющиейся суффиксом п.с.к. с начальным балансом b, длины m
$A_{m,b} = A_{m-1,b+1} + (b > 0?A_{m-1,b-1} : 0)$

6.2.2 Правильные скобочные последовательности

Рассмотрим бинарные деревья (считаем, если потомок один, то задано, левый он или правый, причем деревья в таком случае различны)
Посчитаем количество деревьев из n вершин(обозначим количество за T_n)
Возьмем корень. Пусть в левом поддереве i вершин. Тогда в правом - $n - i - 1$
Тогда $T_n = \sum_{i=0}^{n-1} T_iT_{n-i-1} = \sum_{i=1}^{n} T_{i-1}T_{n-i}$
Тогда $C_n = T_n$
Построим изоморфизм между деревьями и п.с.к.
Тогда дереву с левым поддеревом α и правым поддеревом β п.с.к. "$(\alpha)\beta$"(отсутствие потомка = потомок - пустое дерево)

6.2.3 Деревья с порядком на детях

Рассмотрим деревья с порядком на детях
Если детей несколько, то они имеют порядок, иначе вершины не помечены
Сопоставим деревья и п.с.к.
Пусть у вершины потомки $\alpha_1, \ldots, \alpha_k$. Тогда дереву будет соответствовать "$(\alpha_1 \ldots \alpha_k)$"
Заметим, что в данном случае деревьям соответствуют п.с.к., у которых есть внешние скобки. Тогда $T_n = C_{n-1}$

6.2.4 Разбиения

Пусть p_n - количество разбиений числа n на сумму неубывающей положительной последовательности натуральных чисел $\leq k$
$p_{n,k} = p_{n-k,k} + p_{n,k-1}$(если взяли k в сумму + если не взяли k в сумму)

6.2.5 Разбиения множеств

Количество разбиений множества размера n на k множеств $S_{n,k} = \left\{ \begin{matrix} n \\ k \end{matrix} \right\}$ - число Стирлинга 2 рода
$\left\{ \begin{matrix} n \\ k \end{matrix} \right\} = \left\{ \begin{matrix} n-1 \\ k-1 \end{matrix} \right\} + k \left\{ \begin{matrix} n-1 \\ k \end{matrix} \right\}$
Число Белла B_n - количество разбиений множества размера n на подмножества
$B_n = \sum\limits_k \left\{ \begin{matrix} n \\ k \end{matrix} \right\}$
Числа Стирлинга 1 рода $\left[\begin{matrix} n \\ k \end{matrix} \right]$ - количество перестановок n элементов с k циклами
$\left[\begin{matrix} n \\ k \end{matrix} \right] = \left[\begin{matrix} n-1 \\ k-1 \end{matrix} \right] + (n-1) \left[\begin{matrix} n-1 \\ k \end{matrix} \right]$
$n! = \sum\limits_k \left[\begin{matrix} n \\ k \end{matrix} \right]$

6.3 Перестановки

6.3.1 Циклическое представление

Рассмотрим перестановку P. Представим ее в виде массива чисел. Тогда для всех индексов построим ребро $i \to p[i]$
Такой граф - *граф перестановки*
Этот граф будет являться набором циклов
Тогда каждую перестановку можно задать набором циклов (не единственным образом)
К примеру, перестановку 31425 можно задать набором циклов:
(1342)(5)
(3421)(5)
(5)(2134)

и т.д.
В канонической записи на первое место в цикле ставят наибольший элемент, а циклы сортируют по возрастанию первого элемента
Для нашей перестановки канонической записью будет (4213)(5). При этом скобки в такой записи можно убрать: 42135
Такую запись называют *фундаментальным изоморфизмом*
Набор длин цикров перестановки называется ее *циклическим классом*
Заметим, что при возведении в квадрат циклов нечетной длины меняется порядок элементов в цикле, а при возведении в квадрат циклов четной длины разбиваются на 2 цикла половинной длины

6.3.2 Перестановки как группа

Перестановка является *действием*
Заметим, что к действиям могут быть применены композиции
Композицию перестановок назовем *произведением перестановок*
Обозначим S_n множество перестановок
Тогда композиция $\cdot : S_n \times S_n \to S_n$
$c = b \cdot a \Leftrightarrow c[i] = b[a[i]]$
Перестановки - *группоид*, т.е. множество с одной операцией
Свойства композиций:

1. $(ab)c = a(bc)$ - перестановки - *полугруппа* (т.е. множество с коммутативной операцией)
   Доказательство
   $((ab)c)[i] = (ab)[c[i]] = a[b[c[i]]] = a[(bc)[i]] = (a(bc))[i]$

2. $\exists id : id \cdot a = a \cdot id = a$
   Тогда перестановки - *моноид*
   Доказательство
   $id = [1, 2, 3, 4, \ldots]$

3. $\forall\, a\ \exists\, a^{-1} : a^{-1}a = aa^{-1} = id$
   Инволюция - такие a, что $a^{-1} = a$
   Все перестановки, где длины всех циклов не более 2 - инволюции
   Доказательство
   $a^{-1}[a[i]] = i$
   (все ребра развернуты)

Т.о. перестановки - группа

Конгруэнтность - отношение эквивалентности, согласованное с некоторой операцией

Таблица Кэли - "таблица умножения"для операции в группе

Утверждение Заметим, что в таблице Кэли в каждой строке все элементы различны

Доказательство

Выберем строчку, соответствующую второму операнду a

Пусть в этой строчке $\exists\, b, c: \ ba = ca$

Тогда $baa^{-1} = caa^{-1}$

Тогда $b = c$, ч.т.д.

Тогда таблица Кэли содержит перестановки

Подгруппа группы - группа, полученная выкидыванием из группы некоторых элементов, при котором для всех элементов группе остались обратные им, сохранился нейтральный элемент и результат операции над любыми элементами лежит в группе

Теорема Кэли

Любая конечная группа изоморфна подгруппе группы перестановок

Доказательство

Пусть G - наша конечная группа, $|G| = n$

Рассмотрим S_n

$G = \{g_1, \dots, g_n\}, e = g_1$

Тогда наша таблица Кэли выглядит вот так:

	g_1	g_2	$\cdots$	g_n
g_1	g_1	g_2	$\cdots$	g_n
$\vdots$				
g_i	$g_ig_1 = g_{\pi_1}$	$g_ig_2 = g_{\pi_2}$	$\cdots$	$g_ig_n = g_{\pi_n}$
$\vdots$				
g_j	$g_jg_1 = g_{\sigma_1}$	$g_jg_2 = g_{\sigma_2}$	$\cdots$	$g_jg_n = g_{\sigma_n}$ $\vdots$
g_k	$g_kg_1 = g_{\theta_1}$	$g_kg_2 = g_{\theta_2}$	$\cdots$	$g_kg_n = g_{\theta_n}$

Пусть $e \leftrightarrow id$

Сопоставим g_i и перестановку π, g_j - перестановку σ

Рассмотрим теперь g_{θ_t} :

$g_{\theta_t} = g_kg_t = g_ig_jg_t = g_ig_{\sigma_t} = g_{\pi_{\sigma_t}}$

Из определения композиции $\pi_{\sigma_t} = (\pi\sigma)_t$

Тогда перестановка $\theta = \pi\sigma$

Т.о. мы построили изоморфизм между перестановками и элементами группы

6.3.3 Матричное представление

Возьмем перестановку π
Построим матрицу смежности A'_π графа π:
$A'_\pi[i][\pi_i] = 1$
Тогда произведение $A'_\pi \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{pmatrix}$ будет давать нам нашу перестановку, т.е. к $[1, 2, 3, 4, 5]$ была применена обратная перестановка
Транспонируем матрицу
Теперь умножение $A_\pi = A'^T_\pi$ на столбец будет равносильно применению к этому столбцу нашей перестановки
Композиция $\pi\sigma = A_\pi A_\sigma$

6.4 Подсчет комбинаторных объектов

Определение
Рассмотрим множество действий F и множество элементов A
Тогда *действием на множестве* $\cdot : F \times A \to A$
Пример 1
$F = \sigma \in S_n, a = [a_1, \ldots, a_n] \in A$
Тогда $(\sigma, a) = [a_{\sigma_1^{-1}}, \ldots, a_{\sigma_n^{-1}}]$

Пусть F действует на $A, a, b \in A$
Введем отношение $\sim_F$: $a \sim_F b \Leftrightarrow \exists\, f \in F : a = f \cdot b$
Введем аксиомы отношения эквивалентности, наложив ограничения на F:

1. $\exists\, e \in F :\ e \cdot a = a$ (отсюда $a \sim_F a$)
2. $\forall\, f \in F\ \exists\, f^{-1} :\ f^{-1} \cdot f = e$ (отсюда $a \sim_F b \Leftrightarrow b \sim_F a$)
3. F замкнута по композиции (отсюда $a \sim_F b, b \sim_F c \Rightarrow a \sim_F c$)
4. Потребуем ассоциативность композиции (отсюда F - группа)

Теорема
Если F - группа относительно композиции, то $\sim_F$ - отношение эквивалентности
Определение
Рассмотрим группу G и множество A
Пусть G *действует* на A, если $\exists \cdot : G \times A \to A, e \cdot a = a, (g \circ h) \cdot a = g \cdot (h \cdot a)$
Определение
Классы эквивалентности A относительно $\sim_G$ $A|_{\sim_G}(A/G)$ называются *орбитами*
Определение
Множество неподвижных точек $I_g = \{a \in A : g \cdot a = a\}$
Стабилизатор $a \in A$ $\operatorname{St} a = \{g : g \cdot a = a\}$
Лемма
$$\sum_{g \in G} |I_g| = \sum_{a \in A} |\operatorname{St} a|$$
Доказательство
$$\sum_{g \in G} |I_g| = \sum_{g \in G} \sum_{a \in A : ga = a} 1$$
$$\sum_{a \in A} |\operatorname{St} a| = \sum_{a \in A} \sum_{g \in G : ga = a} 1$$
Теорема (лемма Бернсайда)
$$|A/G| = \frac{\sum_{g \in G} |I_g|}{|G|}$$
Доказательство
$$\begin{array}{c|cccc} & a_1 & a_2 & a_i & a_j \\ \hline & & & & \\ & & & & \end{array}$$ //todo
Определение
Ожерелье $\in N_{n,k}$ - объект, состоящий из k видов элементов и имеющий длину (количество элементов) n, такой, что ожерелье равно полученному из него циклическим сдвигом
$$|N_{n,k}| = \frac{\sum_{i=0}^{n-1} |I_i|}{n} = \frac{\sum_{i=0}^{n-1} k^{\operatorname{cyc}(n,i)}}{n} = \frac{\sum_{i=0}^{n-1} k^{\gcd(n,i)}}{n}$$
//todo что такое циклы
Лемма
$\operatorname{cyc}(n,i) = \gcd(n,i)$
Формула Пойа

$|A/G| = \frac{\sum_{g \in G} k^{\mathrm{cyc}(n,i)}}{|G|}$
//todo что такое циклы