Оглавление

Лекция 2: Разбиения, прямое произведение, нумерация

20.09.2023

Теорема 1. Произведение разбиений существует.

Доказательство. $\mathcal{A}, \mathcal{B}$ — разбиения.

Возьмем все множества вида $C_{ij} = A_i \cap B_j$

- C — измельчение $\mathcal{A}$, так как $\forall C_{ij}\ \exists A_i : C_{ij} \subset A_i$
- аналогично C — измельчение $\mathcal{B}$

Предположим, что F — измельчение, большее C, тогда:

$\forall F_k : \begin{cases} \exists A_i : F_k \subset A_i \\ \exists B_i : F_k \subset B_i \end{cases} \Rightarrow F_k \subset A_i \cap B_j \Rightarrow F_k \subset C_{ij}$ — C наибольшее из измельчений. $\square$

Глава 1

Алгоритмы перебора

1.1 Перебор 0-1 векторов

Будем рассматривать множество B^m всех наборов из m битов, каждый из которых может быть нулем и единицей. Элемент множества B^m — вектор $(0, 0, 1, \ldots, 1)$ длиной m. Количество элементов в множестве (мощность): $|B^m| = 2^m$.

Для того, чтобы создать вычислительный процесс, при котором на каждом шаге будет формироваться новый, не встречавшийся ранее, элемент рассматриваемого множества, достаточно заметить, что существует взаимнооднозначное соответствие между числами из $0 \ldots 2^m - 1$ и наборами 0-1 векторов. Т. е. достаточно первым взять число 0 и его двоичное представление $(0, \ldots, 0)$, а затем просто добавлять по единице, имитируя это на текущем наборе, пока мы не дойдем до набора из одних единиц.

Кроме рассмотренного способа перебора наборов, можно предложить другой алгоритм, который на каждом шаге меняет значение только одной компоненты:

> **Алгоритм.** (Перебор и нумерация 0-1 векторов в порядке минимального изменения)
>
> - создаем 2 набора x и y, каждый из m битов. Первоначально $x = y = (0, 0, 0, 0)$
> - прибавляем к x единицу и фиксируем позицию j, где произошло изменение.
> - изменить j-ую компоненту в наборе $y : y_j = 1 - y_j$
> - вернуть y

> **Пример.** (Рассмотрим на примере $m = 4$)

x	y	j
0000	0000	-
0001	0001	4
0010	0011	3
0011	0010	4
0100	0110	2
0101	0111	4

1.2 Перебор прямого произведения

Рассматриваем множество $M(1:k) = M_1 \times M_2 \times \ldots \times M_k$. Число элементов: $\prod_{i \in 1:k} m_i$, где $m_i = |M_i|$.

Будем считать, что каждое M_i состоит из m_i элементов, которые мы будем нумеровать от 0 до $m_i - 1$. Тогда каждый элемент $M(1:k)$ — последовательность неотрицательных чисел $(r_1, \ldots, r_k), r_i < m_i$

Общая формула перехода от элемента $(r_1, \ldots, r_k)$ к номеру этого элемента:

$$num(r_1, \ldots, r_k) = \sum_{i=1}^{k} \times (\prod_{j=1}^{i-1} m_j)$$

1.3 Перебор перестановок

Рассмотрим множество $T_k = M_1 \times M_2 \times \ldots \times M_k, M_i = \{0, 1, \ldots, i-1\}, |T_k| = k!$. Обозначим множество всех перестановок из k элементов через P_k.

Построим взаимнооднозначное соответствие между T_k и P_k. Возьмем перестановку $(r_1, \ldots, r_k)$ и сопоставим ей элемент $(t_1, \ldots, t_k)$ следующим образом: $\forall i \in 1:k$ найдем число значений, меньших r_i среди $r_{i+1}, \ldots, r_k$ — это число перепишем в качестве t_i.

Пример.

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

Чтобы получить перестановку по записи $(t_1, \ldots, t_k)$, нужно помнить множество значений S_i, которые могут быть в перестановке на i-ом месте. Так, $S_1 = 1:8, t_1 = 3$ означает, что $r_1 = 4$. Далее $S_2 = 1:3 \cup 5:8, t_2 = 6$ значит, что $r_2 = 8$

Замечание. Если использовать отображение из примера при переборре, то перестановки будут идти в лексикографическом порядке Это значит, что:

$(r_1, \ldots, r_k)$ предшествует $(R_1, \ldots, R_k) \Leftrightarrow$ начала этих перестановок совпадают до i индекса, а далее $r_i < R_i$

Замечание. Очевидно, что если факториальная запись $(t_1, \ldots, t_k)$ лексикографически предшествует другой, то порядок верен и для соответствующих перестановок.

Алгоритм. (Перебор перестановок в лексикографическом порядке)

1. в заданной перестановке $(r_1, \ldots, r_k)$ найдем наибольший суффикс $(r_t, \ldots, r_k)$, в котором элементы расположены по убыванию.
2. выбрать в $(r_t, \ldots, r_k)$ элемент, следующий по велечине после r_{t-1} и поставить его на r_{t-1}. Оставшиеся эелменты, включая r_{t-1} расположить за ним в порядке возрастания.

Пример.

3	4	2	1	7	8	9	5	**6**
3	4	2	1	7	8	**9**	**6**	**5**
3	4	2	1	7	9	5	6	**8**
3	4	2	1	7	9	5	**8**	**6**
3	4	2	1	7	9	6	5	**8**
3	4	2	1	7	9	6	**8**	**5**
3	4	2	1	7	9	8	5	**6**
3	4	2	1	7	**9**	**8**	**6**	**5**
3	4	2	1	8	5	6	7	**9**