Генерация комбинаторных объектов

Генерация сочетаний в лексикографическом порядке

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

Порождение сочетаний

Сочетанием из n элементов по k называется неупорядоченная выборка k элементов из заданных n элементов.

Будем предполагать, что $A = \{1,2, \dots, n\}$. Произвольное сочетание из n по k удобно представить в виде конечной последовательности длины k из чисел, упорядоченных по возрастанию слева направо.

Порождение сочетаний

При $n = 5$ и $k = 3$ последовательность всех сочетаний в лексикографическом порядке следующая:

$$123,\ 124,\ 125,\ 134,\ 135,\ 145,\ 234,\ 235,\ 245,\ 345.$$

Начальное значение и остановка

Первый элемент в лексикографическом порядке есть сочетание

$$(1,2,\ldots,k),$$

а последний

$$(n-k+1,n-k+2,\ldots,n-1,n).$$

Начальное значение и остановка

Определим по данному сочетанию $a = (a_1, \dots, a_k)$ вид непосредственно следующего сочетания $b = (b_1, \dots, b_k)$.
Такое сочетание b получается в результате нахождения самого правого элемента a_m, который ещё не достиг своего возможного максимального значения,
его увеличения на 1, а затем присвоения всем элементам справа от этого элемента новых возможных наименьших значений.

Трансформация текущего сочетания

Для этого необходимо найти самый правый элемент a_m такой, что чисел больших, чем $a_m + 1$ найдётся по крайней мере $k - m$ штук

Таким образом,

$$b = (a_1, \dots, a_{m-1}, a_m + 1, a_m + 2, \dots, a_m + k - m + 1),$$

где

$$m = \max\{i : a_i < n - k + i, 1 \le i \le k\}.$$

Порождение сочетаний

$a_0 = -1;$
for $i = 1$ **to** k **do** $a_i = i;$
$m = 1;$
while $m \neq 0$ **do**
begin
  write$(a_1, \dots, a_k);$
  $m = k;$
  while $a_m = n - k + m$ **do** $m = m - 1;$
  $a_m = a_m + 1;$
  for $i = m + 1$ **to** k **do** $a_i = a_{i-1} + 1;$
end.

Порождение размещений

Так как размещения – это упорядоченные k-элементные подмножества множества из n элементов (упорядоченные сочетания из n по k), то алгоритм их порождения можно получить путем комбинации алгоритмов порождения сочетаний и перестановок.

Порождение подмножеств множества

Порождение подмножеств множества

$$\{a_1, \dots, a_n\}$$

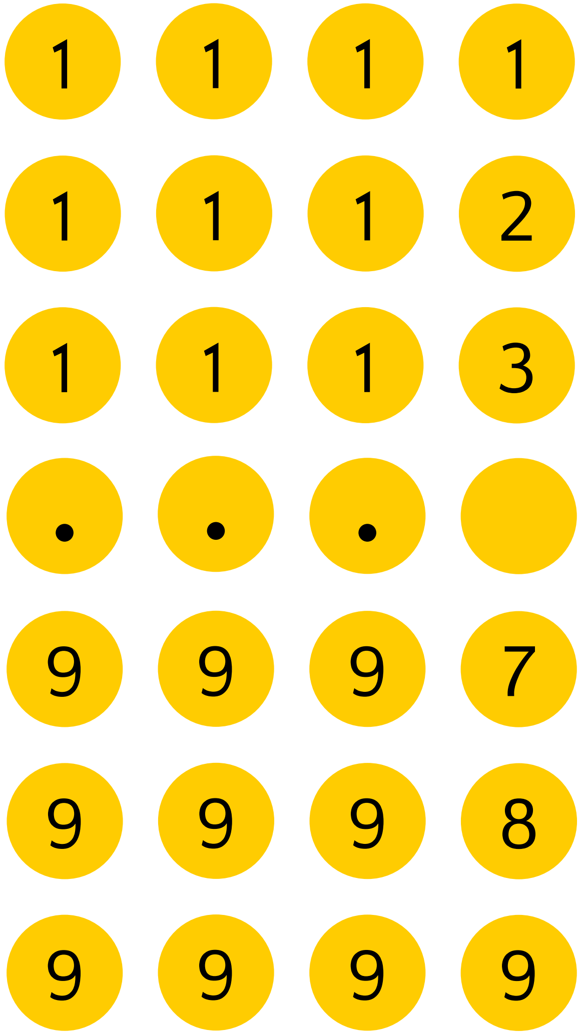
эквивалентно порождению n-разрядных двоичных наборов a_i, принадлежащих подмножеству, если и только если i-ый разряд равен единице.

Таким образом, задача порождения всех подмножеств множества сводится к задаче порождения всех возможных двоичных последовательностей длины n.

Счёт в системе счисления с основанием 2

for $i = 0$ **to** n **do** $b_i = 0$;
while $b_n \neq 1$ **do**
begin
  write $(b_{n-1}b_{n-2}, \dots, b_0)$;
  $i = 0$;
  while $b_i = 1$ **do**
  begin
    $b_i = 0$;
    $i = i + 1$;
  end;
  $b_i = 1$;
end.

Порождений конфигураций с повторениями



Размещения с повторениями

Имеются предметы n различных видов $a_1, a_2, \dots, a_n$.
Из них составляют всевозможные расстановки длины k.
Такие расстановки называются размещениями
с повторениями из n по k (элементы одного вида
могут повторяться).
Общее число размещений с повторениями
из n по k равно

$$\overline{A_n^k} = n^k.$$

Генерация размещений с повторениями

Порождение множества всех размещений с повторениями длины k из элементов

$$\{a_0, a_1, \dots, a_{n-1}\}$$

эквивалентно генерации множества k-разрядных чисел в системе счисления с основанием n: на r-м месте в размещении будет располагаться элемент a_i, если цифра в r-м разряде соответствующего числа равна i.

Генерация размещений с повторениями

Например, для $k = 2, n = 3$ получаем

00, 01, 02, 10, 11, 12, 20, 21, 22

откуда имеем следующие подмножества

$$(a_0, a_0), (a_0, a_1), (a_0, a_2), (a_1, a_0),$$
$$(a_1, a_1), (a_1, a_2), (a_2, a_0), (a_2, a_1), (a_2, a_2).$$

Счёт в системе счисления с основанием $\boldsymbol{n}$ для порождения всех $\boldsymbol{k}$-разрядных наборов

for $i = 0$ **to** k **do** $b_i = 0$;
while $b_k \neq 1$ **do**
begin
  write$(b_{k-1} b_{k-2} \dots b_0)$;
  $i = 0$;
  while $b_i = n - 1$ **do**
  begin
    $b_i = 0$;
    $i = i + 1$;
  end;
 $b_i = b_i + 1$
end

Сочетания с повторениями

Имеются предметы n различных видов. Число элементов каждого вида неограниченно. Сколько существует расстановок длины k, если не принимать во внимание порядок элементов?

Сочетания с повторениями

Такие расстановки называют сочетаниями с повторениями.

Количество сочетаниями с повторениями равно

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

Генерация сочетаний с повторениями

Перестановки с повторениями

Задача формулируется следующим образом.
Имеются предметы k различных видов.
Сколько существует перестановок из
n_1 элементов первого типа,
n_2 элементов второго типа,
…
n_k элементов k-то типа?

Генерация перестановок с повторениями

$$\alpha_1 = (1, 1, 1, 2, 3, 3, 4, 4, 4, 4)$$
$$\alpha_2 = (1, 1, 1, 2, 3, 4, 3, 4, 4, 4)$$
$$\alpha_3 = (1, 1, 1, 2, 3, 4, 4, 3, 4, 4)$$
$$\alpha_4 = (1, 1, 1, 2, 3, 4, 4, 4, 3, 4)$$
$$\alpha_5 = (1, 1, 1, 2, 3, 4, 4, 4, 4, 3)$$
$$\alpha_6 = (1, 1, 1, 2, 4, 3, 3, 4, 4, 4)$$
$$\alpha_7 = (1, 1, 1, 2, 4, 3, 4, 3, 4, 4)$$

$$\alpha'_1 = (1, 1, 1, 2, 3, 4, 4, 4, 4, 3)$$
$$\alpha'_2 = (1, 1, 1, 2, 3, 4, 4, 4, 4, 3)$$
$$\alpha'_3 = (1, 1, 1, 2, 3, 4, 4, 4, 4, 3)$$
$$\alpha'_4 = (1, 1, 1, 2, 3, 4, 4, 4, 4, 3)$$
$$\alpha'_5 = (1, 1, 1, 2, 4, 4, 4, 4, 3, 3)$$
$$\alpha'_4 = (1, 1, 1, 2, 4, 3, 4, 4, 4, 3)$$

Лабораторная работа 1

1.2. [# 25] Задача о назначении. Для выполнения проекта необходим набор навыков S. Есть группа людей, каждый из которых владеет некоторыми из этих навыков. Необходимо сформировать наименьшую подгруппу, достаточную для выполнения задания, т.е. включающую в себя носителей всех необходимых навыков.

Задача о назначении

Требуется написать функцию

```
bool NextCombinations(size_t dim, std::vector<size_t>& state)
```

преобразующую текущий вектор сочетаний state в вектор сочетаний следующий в лексикографическом порядке.

Функция должна возвращать true, если следующее в лексикографическом порядке сочетание существует и false – в противном случае.

Лабораторная работа 1

1.2. [# 25] Задача об укладке рюкзака. Есть n различных предметов. Каждый предмет с номером i, где $i = 1, \dots, n$, имеет заданный положительный вес w_i и стоимость c_i. Нужно уложить рюкзак так, чтобы общая стоимость предметов в нём была наибольшей, а вес – меньше заданного числа $W_{\max}$.
Форма предметов значения не имеет.

Задача об укладке рюкзака

Требуется написать функцию

```
bool NextOccurrence(std::vector<size_t>& occurrence)
```

преобразующий текущий индикаторный вектор в следующий.
Функция должна возвращать true, если следующий вектор размерности occurrence.size() существует
и false – в противном случае.

Понижающие коэффициенты

Сдача работы без тестов – коэффициент 0.5.
К задачам с графами необходимы изображения графов для каждого теста.
Тесты не покрывают всевозможные ситуации от 0.5 до 1.0.

Сдача в течении недели после выдачи – коэффициент 1.0.
Сдача в течении двух недель после выдачи – коэффициент 0.8.
Сдача через две недели после выдачи – коэффициент 0.6.