第四章：生成排列和组合

4.1 生成排列
4.2 排列中的逆序
4.3 生成组合
4.4 生成 r 子集

组合数学

(1) 存在：满足一定条件配置的存在性.

(2) 计数：计算出满足条件配置的数目.

(3) 算法：构造所有配置的算法.

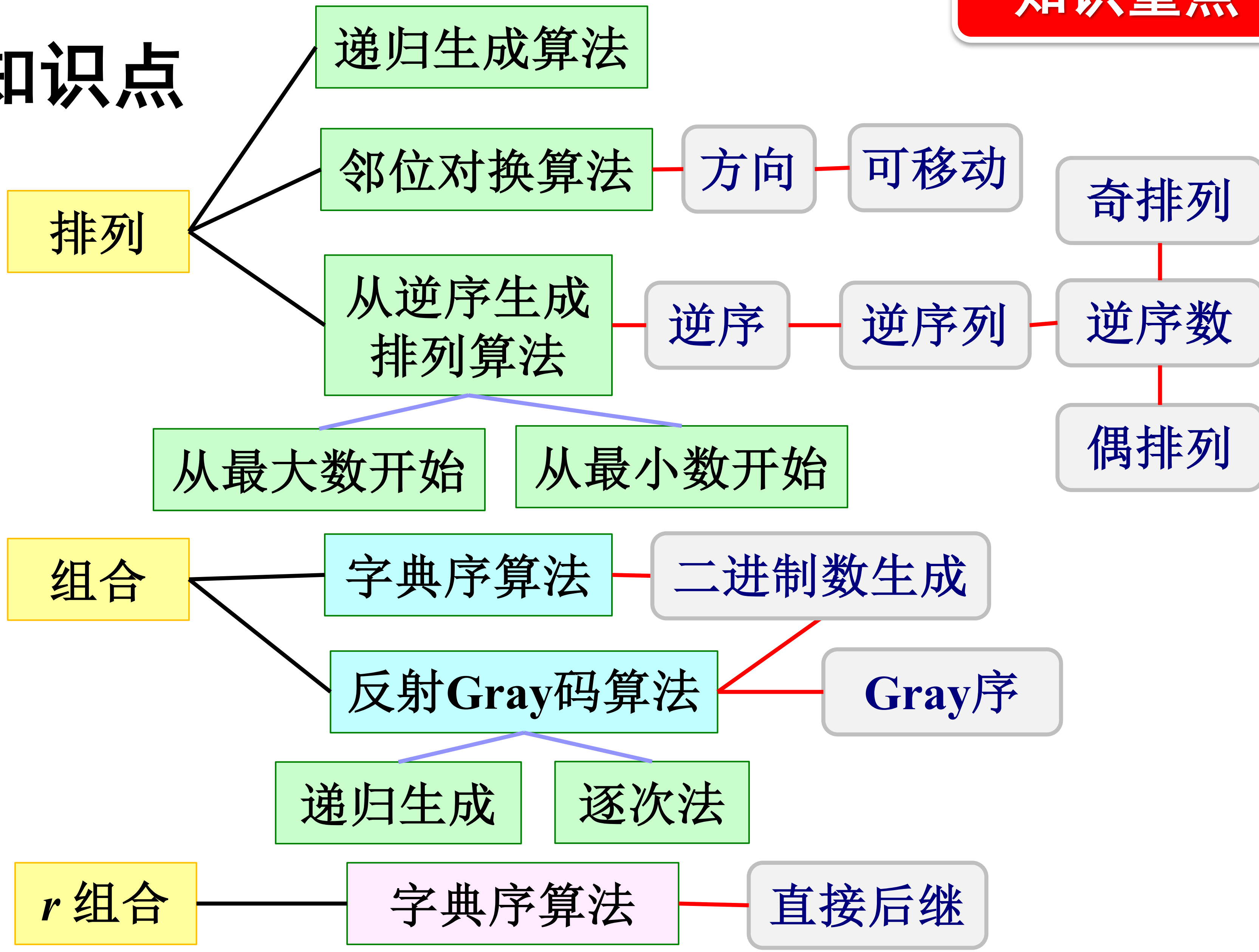
(4) 优化：优化算法.

主要内容

- 排列生成算法：
  - 递归生成算法
  - 邻位对换算法
  - 从逆序生成排列算法
- 组合生成算法
  - 字典序
  - 反射Gray码
  - 基于字典序的r组合生成算法

知识重点

知识点



排列
递归生成算法
邻位对换算法
方向
可移动
从逆序生成排列算法
逆序
逆序列
逆序数
奇排列
偶排列
从最大数开始
从最小数开始
组合
字典序算法
二进制数生成
反射Gray码算法
Gray序
递归生成
逐次法
r 组合
字典序算法
直接后继

第四章：生成排列和组合

4.1 生成排列

4.2 排列中的逆序

4.3 生成组合

4.4 生成 r 子集

一种最为初级的“黑客”技术

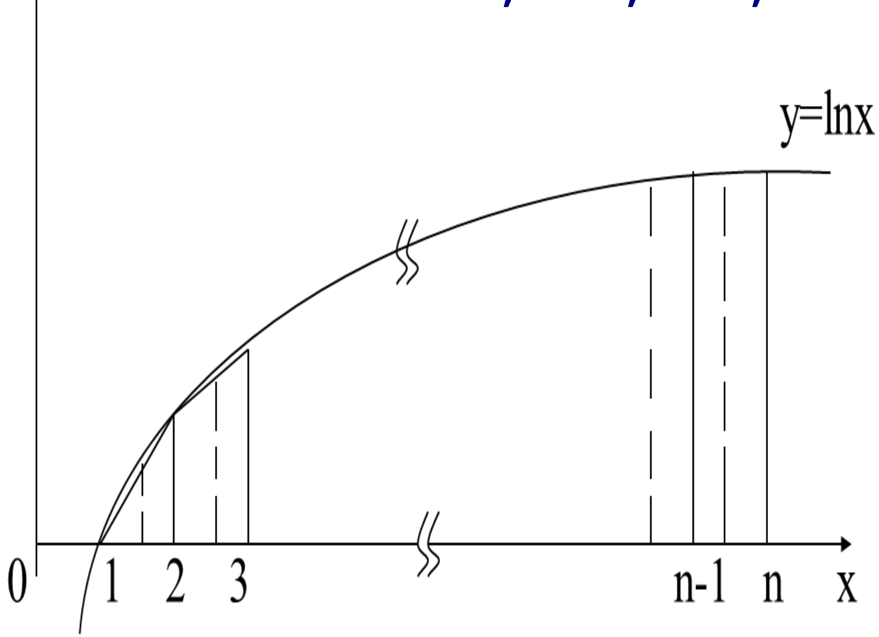
- **穷举攻击：** 最初的DES密码是40位二进制数。编一个程序，尝试所有可能的密码。要求：
  - 无重复、无遗漏
  - 尽量少的存储空间
  - 尽可能简单操作。
- 如果具有一些“预先知识”，在一些特点字符里选取，如何设计算法？（如字典攻击）

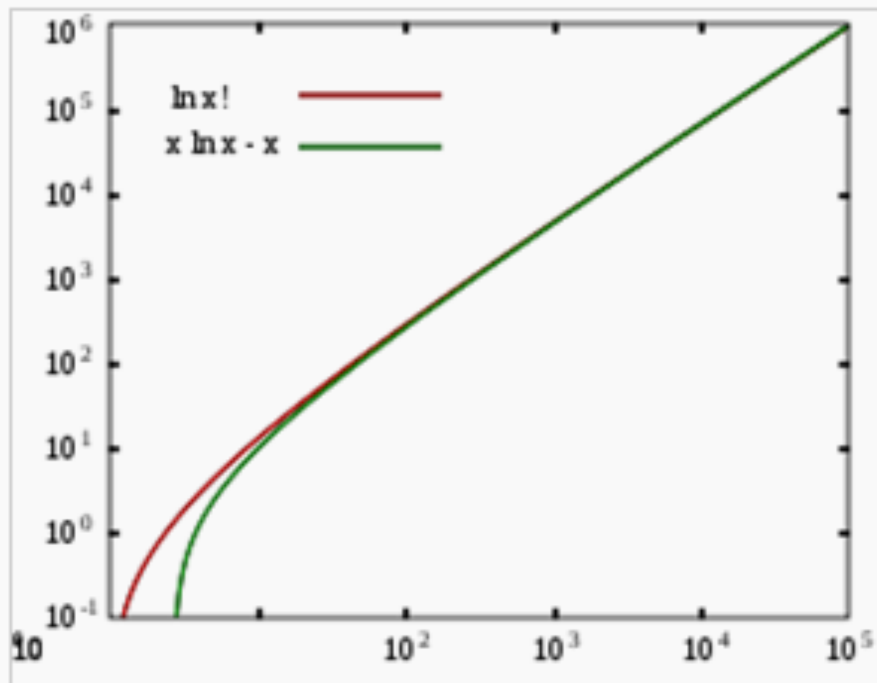
4.1 生成排列

- **Stirling近似公式**

$$n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$$

15! >100,000,000,000





当n增加时，($\ln n!$)与o ($n \ln n - n$)之比趋于1。

排列生成算法

- 生成{1, 2,…, n}的所有排列的算法
  - 算法输出结果为一个表
  - 表包含了{1, 2, …, n}的所有排列
  - 每个排列只出现一次

三种排列生成算法

- 递归生成算法
- 邻位对换算法
- 从逆序生成排列算法

递归生成算法

- **S.M.Johnson (1963)/ H.F.Trotter(1962)**

{1， 2， 3， 4， 5}　　**{1， 2， 3， 4}**

3, 2, 4, 1, 5　　3, 2, 4, 1　　□3, □ 2, □ 4, □ ,1 □

3, 2, 4, 5, 1

3, 2, 5, 4, 1

3, 5, 2, 4, 1

5, 3, 2, 4, 1

- 观察1：将整数 *n* 从{1, 2, …, *n*}的一个排列中删除后，得到一个{1, 2,…, *n*–1}的排列。
- 观察2：同一个*n*-1排列可以从不同的（*n*个）*n*排列生成
- 观察3：从{1, 2,…, *n*–1}的一个排列可生成 *n* 个{1,2,…,*n*}的排列

算法基本思想

- 对集合{1, 2, …, n-1}的每一个排列进行如下操作（一共(n-1)!个排列）：
  - 把 n 插入到首、尾和任两个数的中间共 n 个位置，产生集合{1, 2, …, n}的 n 个排列
- 从而产生 $n\times(n-1)!=n!$个集合{1, 2, …, n}的排列。

n=5时　□3, □ 2, □ 4, □ ,1 □

{1}的所有排列 → {1, 2}的所有排列 → {1, 2, 3}的所有排列 → … → {1,2,…,n}的所有排列

算法描述

n=1: 1

n=2: 1 2

2 1

n=3: 1 2 3

1 3 2

3 1 2

3 2 1

2 3 1

2 1 3

n=4: 1 2 3 4

1 2 4 3

1 4 2 3

4 1 2 3

…

2 3 1 4

2 3 4 1

2 4 3 1

4 2 3 1

4 2 1 3

2 4 1 3

2 1 4 3

2 1 3 4

n=5, …

当生成的排列为 **213…*n*** 时，算法结束，生成全部排列。

算法分析

- 可归纳验证该算法生成的**最后一个排列是213…*n***。

算法特点：

- 生成{1, 2, …, n }的排列算法**需要存储所有 {1, 2, …, n-1}的排列**，因此，需要巨大的存储空间。
- 算法空间复杂度太高！

观察：

注意：交换两个相邻的数

n=4:

1	2	3	4
1	2	4	3
1	4	2	3
4	1	2	3

…

2	3	1	4
2	3	4	1
2	4	3	1
4	2	3	1
4	2	1	3
2	4	1	3
2	1	4	3
2	1	3	4

什么条件下进行邻位互换？

邻位对换算法

- 对任一给定整数 k, 其上加一个箭头表示移动方向：$\overrightarrow{k}$ 或 $\overleftarrow{k}$ 。
- 对于集合$\{1, 2, \ldots, n\}$的任一个排列，其中每一个整数都有一个箭头指出其移动方向,

如果整数 k 的箭头指向与其相邻但比它小的整数,
则称 k 是可移动（活动）的.

例：序列 $\overrightarrow{2}\,\overrightarrow{6}\,\overrightarrow{3}\,\overleftarrow{1}\,\overrightarrow{5}\,\overrightarrow{4}$ 那几位是活动的？

只有3、5、6是活动的。

邻位对换算法

- 对任一给定整数 k, 其上加一个箭头表示移动方向：$\vec{k}$ 或 $\overleftarrow{k}$ 。
- 对于集合{1, 2, …, n}的任一个排列，其中每一个整数都有一个箭头指出其移动方向,

如果整数 k 的箭头指向与其相邻但比它小的整数, 则称 k 是可移动（活动）的.

例：序列 $\vec{2}\ \vec{6}\ \vec{3}\ \overleftarrow{1}\ \vec{5}\ \vec{4}$ 那几位是活动的？

只有3、5、6是活动的。

注：(1) 在任意序列中，1 绝对不可能是活动，是否正确？ 正确！

(2) 在{1, 2, …, n}元素构成的任意序列中，n 是否一定是活动的？

除去以下两种情况：

- n是第一个整数而它的箭头 指向左边：

$$\overleftarrow{n}\ldots$$

- n是最后一个整数而它的箭头指向右边：

$$\ldots\overrightarrow{n}$$

邻位对换算法

生成{1, 2, …, n}的排列算法:

1. 初始：$\overleftarrow{1}\ \overleftarrow{2}\ \ldots\ \overleftarrow{n}$;

2. while 存在活动整数时， do

   (1) 求出最大的活动整数 *m*

   (2) 交换 *m* 和其箭头指向的相邻整数的位置

   (3) 改变所有满足*p*>*m*的整数 *p* 的箭头方向。

3. 不存在活动整数时，算法结束。

算法例子

2. 存在活动整数时， do

(1). 求出最大的活动整数*m*

(2). 交换*m*和其箭头指向的相邻整数的位置

(3). 改变所有满足*p*>*m*的整数 *p* 的箭头方向

***n*=4的算法描述**

没有比4大的整数

***m* = 4**

← ← ← ←
1 2 3 4

交换

← ← ← ←
1 2 4 3

算法例子

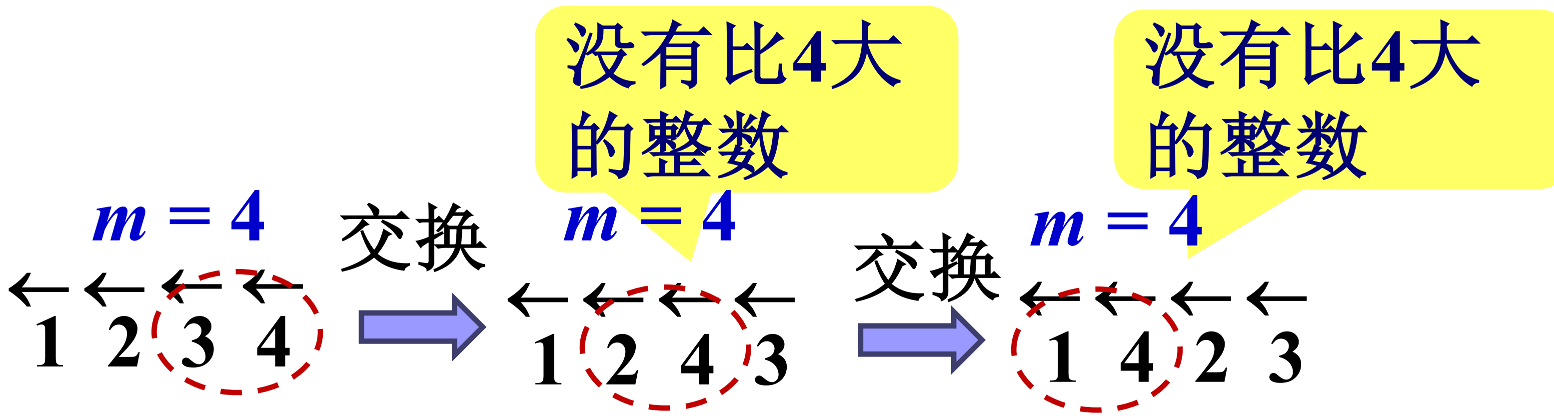
2. 存在活动整数时， do

(1). 求出最大的活动整数m

(2). 交换m和其箭头指向的相邻整数的位置

(3). 改变所有满足$p>m$的整数 p 的箭头方向

n=4的算法描述



算法例子

2. 存在活动整数时，do

(1). 求出最大的活动整数m

(2). 交换m和其箭头指向的相邻整数的位置

(3). 改变所有满足$p>m$的整数 p 的箭头方向

n=4的算法描述

没有比4大的整数

没有比4大的整数

$m = 4$ ← ← ← ← 1 2 3 4 交换 ⇒ $m = 4$ ← ← ← ← 1 2 4 3 交换 ⇒ $m = 4$ ← ← ← ← 1 4 2 3

交换 ⇒ $m = 3$ ← ← ← ← 4 1 2 3 交换 ⇒ ← ← ← ← 4 1 3 2 ⇒ $m = 4$ → ← ← ← 4 1 3 2

4>3

交换 ⇒ ← → ← ← 1 4 3 2 交换 ⇒ ← ← → ← 1 3 4 2 … ⇒ ← ← → → 2 1 3 4

没有比4大的整数

没有活动整数，算法结束

排列生成算法

- 递归生成

{1}的所有排列 → {1, 2}的所有排列 → {1, 2, 3}的所有排列 → … → {1, 2,…, n}的所有排列

- 邻位对换算法
  - 从排列 123…n 开始，生成所有 n 阶排列
  - 活动：箭头指向比其小的整数
  - 邻位对换

- 结论：两种算法生成的排列顺序一致

第四章：生成排列和组合

4.1 生成排列

4.2 排列中的逆序

4.3 生成组合

4.4 生成 r 子集

4.2 排列的逆序

(逆序)：令$i_1\, i_2\, ,\ldots,\, i_n$ 是集合$\{1, 2, \ldots, n\}$的一个排列，如果$0 \le k < l \le n$，且$i_k > i_l$，称数对（i_k, i_l）是排列的一个逆序。

例: (1) 排列 31524 有几组逆序？

(3, 1)，(3, 2)，(5, 2)，(5, 4)

(2) 排列 361245 有几组逆序？

(3, 1)，(3, 2)，(6, 1)，(6, 2)，(6, 4)，(6, 5)

(3) 唯一没有逆序的排列 $1\ 2\ 3\ \ldots\ n$

对于{1, 2, …, n}上的一个排列，令 a_j 表示第二元是 j 的逆序的数量，即 a_j 是排列中先于整数 j 并大于 j 的整数的个数，用于度量 j 的反序程度，称为 j 的逆序数。

例: (1) 排列 31524

逆序: (3, 1), (3, 2), (5, 2), (5, 4)

$a_1=1$,
$a_2=2$,
$a_3=0$,
$a_4=1$,
$a_5=0$

(2) 排列 361245

逆序: (3, 1), (3, 2), (6, 1), (6, 2), (6, 4), (6, 5)

$a_1=2$,
$a_2=2$,
$a_3=0$,
$a_4=1$,
$a_5=1$,
$a_6=0$

（逆序列）：令 a_j 表示排列 $i_1\ i_2\ ,\ldots,\ i_n$ 中数 j 的逆序数，称 $a_1\ a_2\ ,\ldots,\ a_n$ 为排列 $i_1\ i_2\ ,\ldots,\ i_n$ 的逆序列

例：排列 31524

逆序: (3, 1)， (3, 2) ， (5, 2)， (5, 4)

$a_1=1,\ a_2=2,\ a_3=0,\ a_4=1,\ a_5=0$

逆序列：1 2 0 1 0

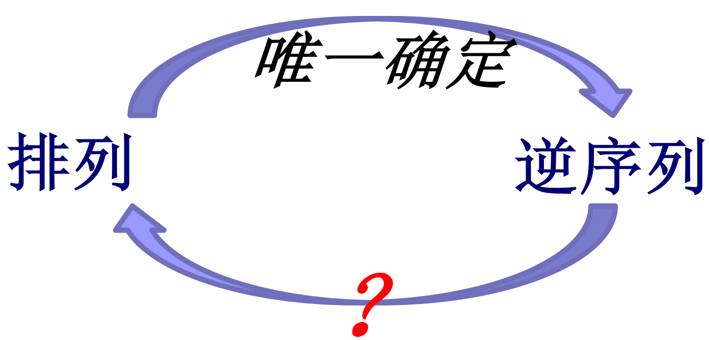
- 结论：一个排列唯一确定一个逆序列。
- 问题：一个逆序列是否唯一确定一个排列？

逆序的性质

- 性质1：排列 $i_1\, i_2\, ,\ldots,\, i_n$的逆序列 $a_1a_2\, ,\ldots,a_n$ 满足：

  $0 \le a_1 \le n-1$， $0\le a_2 \le n-2$， …， $0\le a_{n-1} \le 1$, $a_n = 0$ （1）

- 性质2：满足条件（1）的序列 $a_1a_2\ldots a_n$ 有$n!$个。



逆序列与排列一一对应？？？

基于“逆序”的排列生成算法

定理4.2.1: 令$b_1, b_2,\ldots, b_n$为满足

$$0\le b_1 \le n\text{-}1,\ 0\le b_2 \le n\text{-}2,\ldots,\ 0\le b_{n\text{-}1} \le 1,\ b_n=0$$

的整数序列，那么存在集合$\{ 1, 2, \ldots, n \}$的唯一一个排列，满足它的逆序列为$b_1\ b_2 \ldots b_n$。

证明思路：构造性证明方法，证明过程给出有效算法。

- 从最大数开始
- 从最小数开始

算法描述 (从最大数开始)

由任一个逆序列$b_1, b_2, \ldots, b_n=0$ ($0 \le b_1 \le n$-1，$0 \le b_2 \le n$-2,…, $0 \le b_{n-1} \le 1$，$b_n=0$)，构造一个排列

n：写出整数 n（从最大数开始）

$n-1$：若 $b_{n-1}=0$, 则n-1必在n的前面： n-1, n

若$b_{n-1}=1$：则n-1必在n的后面： n, n-1

$n-2$：若 $b_{n-2}=0$, 则n-2必在前面排列的前面 n-2, n-1, n /n-2, n, n-1

若 $b_{n-2}=1$, 则n-2必在前面排列的两个数中间 n-1, n-2, n

n, n-2, n-1

若 $b_{n-2}=2$, 则n-2必在前面排列的后面 n-1, n, n-2/n, n-1, n-2

$n-k$：若 $b_{n-k}=0$，则n-k必在前面排列的前面。

若 $b_{n-k}=1$，则n-k必在前面排列的前两个数中间。

…

若 $b_{n-k}=k$，则n-k必在上一步得到的排列的后面。

1： b_1，把1放在上一步得到的排列的第 b_1个数的后面。

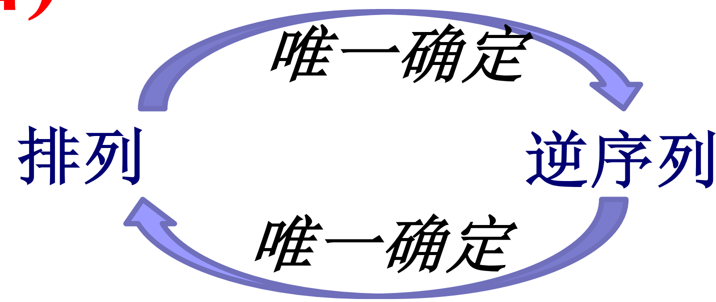
算法举例(从最大数开始)

(1) 已知{1, 2, …, 8}的一个排列的逆序列为 5 3 4 0 2 1 1 0, 确定此排列。

8:	8	0
7:	8 7 ……	1
6:	8 6 7 ……	1
5:	8 6 5 7 …	2
4:	4 8 6 5 7 ……	0
3:	4 8 6 5 3 7 ……	4
2:	4 8 6 2 5 3 7 ……	3
1:	4 8 6 2 5 1 3 7 …	5

(2) 已知{1,2,…,8}的一个排列的逆序列为: 4 6 3 2 2 1 1 0, 确定此排列。86431572

算法描述(从最大数开始)



由算法可知:

- 从n, n-1, …, 2, 1每一步都根据b_1, b_2,…, b_n 唯一确定1, 2,…n在排列中的位置。
- 缺点：算法使得整数1, 2, …, n的相对位置保持固定，但每个整数的位置一直要到最后才能确定。

算法描述(从最小数开始)

由任一个逆序列$b_1, b_2, \ldots, b_n=0$ ($0 \le b_1 \le n$-1, $0\le b_2 \le n$-2,…, $0\le b_{n-1} \le 1$, b_n=0)，构造一个排列:

设有n个位置，标记为1, 2, …, n

1	2	3	4	5	6	7	8	…	n

1: 把 1 放在b_1+1位置上 (1前面有b_1数大于1);

2: 把2放在第b_2+1个空位置上 (2前面有b_2个数大于2) ;

…

k: 把 k放在第b_k+1个空位置上 (k前面有b_k个数大于k) ;

…

n: 把 n放在余下的空位置上。

算法唯一地确定1, 2, …, n在排列中的位置

算法举例

逆序列：5 3 4 0 2 1 1 0

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

1： 5＋1

2： 3＋1

3： 4＋1

4： 0＋1

5： 2＋1

6： 1＋1

7： 1＋1

8： 0+1

算法举例

- 已知{**1, 2 , …, 8**}的一个排列的逆序列为 **3 5 2 4 0 0 1 0**,确定此排列。

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

奇排列、偶排列

- 逆序个数为奇数的排列称为奇排列；
- 逆序个数为偶数的排列称为偶排列。

n阶矩阵 $A=[a_{ij}]$ $(i, j=1, 2, .., n)$, 其行列式为

$$\det(A)=\Sigma\varepsilon(i_1i_2\ldots i_n)a_{i_1}a_{i_2}\ldots a_{n_{in}},$$

其中，$\varepsilon(i_1i_2\ldots i_n)$是$i_1i_2\ldots i_n$的符号(偶排列为+1，奇排列为-1)

应用：排序

已知排列$i_1, i_2, \ldots, i_n$的逆序列为$b_1 b_2 \ldots b_n$，且 $k = b_1 + b_2 + \ldots + b_n$为逆序数。

则可以通过 k次交换相邻两个数，转化为$1\ 2\ \ldots\ n$。

例：将361245转换为123456 (逆序列为220110)

3 6 1 2 4 5

3 1 6 2 4 5

1 3 6 2 4 5

1 3 2 6 4 5

1 2 3 6 4 5

1 2 3 4 6 5

1 2 3 4 5 6

15 puzzles

1	2	3	4
5	6	7	8
9	10	11	12
13	15	14	

向上、下、左、右划动小方块

?

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

Sam Loyd

Born	Samuel Loyd January 30, 1841 Philadelphia, United States
Died	April 11, 1911 (aged 70)
Known for	Chess, puzzles, mathematical games

如何用逆序数来证明以上情况无解?

参考资料

- Knuth, Donald (2004), "A Draft of Section 7.2.1.2: Generating All Permutations", ***The Art of Computer Programming, Pre-Fascicle 2B***, http://www-cs-faculty.stanford.edu/~uno/fasc2b.ps.gz .

Donald E. Knuth 高德纳

小结

- 学习了3种不同的排列生成算法：
  - 递归方法。需要存储所有排列。
  - 邻位对换算法。无需存储排列，但只能从开始位置按顺序列举。
  - 逆序生成算法。无需存储排列，可以从指定的排列按序生成。
- 算法各有特点，可以用于不同环境。

第四章：生成排列和组合

4.1 生成排列

4.2 排列中的逆序

4.3 生成组合

4.4 生成 r 子集

主要内容

- 生成组合算法
  - 压缩序
  - 反射Gray序

4.3 生成组合

- n元集合$S=\{x_{n-1}, \ldots, x_1, x_0\}$的所有组合（子集）共有$2^n$个。（$S$的幂集$2^S$）。
- 设计一个算法将 S 的所有组合列举出来。
  - 没有重复
- 特征函数 χ_A: $S \to \{0, 1\}$，$A \subseteq S$

$$对任意x \in S，\chi_A = \begin{cases} 1, & 若x \in A \\ 0, & 若x \notin A \end{cases}$$

注意: 长度为n的二进制数也是2^n个，两者有何联系？

- n元集合$S=\{x_{n-1}, x_{n-2}, \ldots, x_0\}$的组合与长度为$n$的二进制数一一对应

$$\begin{array}{ccccc} x_{n-1} & x_{n-2} & \cdots & x_1 & x_0 \\ a_{n-1} & a_{n-2} & \cdots & a_1 & a_0 \in \{0,1\} \end{array}$$

- 用二进制 $a_{n-1}a_{n-2}\ldots a_0$ 表示 S 的一个组合$\{x_{i_1}, x_{i_2}, \ldots, x_{i_k}\}$，其中$n$-1≥$i_1$>$i_2$>…>$i_k$≥0:

$$a_{i_j}=1\ (j\in[1, k])，其他位置为0$$

例： $S=\{x_7, x_6, \ldots, x_1, x_0\}$的一个组合$\{x_7, x_5, x_1\}$对应的二进制数为 **10100010**

a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0
1	0	1	0	0	0	1	0

- n元集合$S=\{x_{n-1}, x_{n-2}, \ldots, x_0\}$的组合与长度为$n$的二进制数一一对应

$$
\begin{array}{ccccc}
x_{n-1} & x_{n-2} & \cdots & x_1 & x_0 \\
a_{n-1} & a_{n-2} & \cdots & a_1 & a_0 \in \{0,1\}
\end{array}
$$

- 用 S 的组合来表示$[0, 2^{n-1}]$中的 一个整数的二进制表示: 1 所在的位置对应的元素包含在组合中。

例： $n=7$， 整数29 $\in[1, 2^7]$ 的二进制表示为：0011101, 29对应的组合为 $\{x_4, x_3, x_2, x_0\}$

x_6	x_5	x_4	x_3	x_2	x_1	x_0
0	0	1	1	1	0	1

- 如何生成$S=\{x_{n-1}, x_{n-2}, \ldots, x_0\}$的所有$2^n$个组合？
  - 按从小到大的顺序写出0 到 2^n-1的所有数的二进制形式
  - 每次使用二进制数的加法加 1

n=4时，$a_3a_2a_1a_0$, { x_3, x_2, x_1, x_0}的子集

0	0 0 0 0	Φ
1	0 0 0 1	x_0
2	0 0 1 0	x_1
3	0 0 1 1	x_1, x_0
4	0 1 0 0	x_2
5	0 1 0 1	x_2, x_0
6	0 1 1 0	x_2, x_1
7	0 1 1 1	x_2, x_1, x_0
8	1 0 0 0	x_3
9	1 0 0 1	x_3, x_0
10	1 0 1 0	x_3, x_1
11	1 0 1 1	x_3, x_1, x_0
12	1 1 0 0	x_3, x_2
13	1 1 0 1	x_3, x_2, x_0
14	1 1 1 0	x_3, x_2, x_1
15	1 1 1 1	x_3, x_2, x_1, x_0

$\{x_0\}$的所有组合

$\{x_1, x_0\}$的所有组合

$\{x_2, x_1, x_0\}$的所有组合

- 当 $j<n-1$时，$\{x_j, \ldots, x_1, x_0\}$ 的所有组合都在至少含有 $\{x_{n-1},\ldots, x_{j+1}\}$中一个元素的组合的前面

——子集的压缩序

生成={$x_{n-1}, x_{n-2}, \ldots, x_0$}的所有$2^n$个组合的二进制算法

	$a_3a_2a_1a_0$
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1
10	1 0 1 0
11	1 0 1 1
12	1 1 0 0
13	1 1 0 1
14	1 1 1 0
15	1 1 1 1

1. 初始： $a_{n-1}\ldots a_1a_0 = 0\ldots00$
2. 当 $a_{n-1}\ldots a_1a_0 \neq 1\ldots11$时，执行以下操作：
   (1) 求出使得$a_j=0$的最小整数 j
   (2)用1替换 a_j 并用 0 替换每个$a_{j-1},\ldots,a_0$。
3. 当$a_{n-1}\ldots a_1a_0 = 1\ldots11$时算法结束。

二进制加法

算法按自然二进制数顺序生成， 称为n元组字典序。

问题：

- 组合$\{x_6, x_4, x_2, x_1, x_0\}$ 的下一个组合是什么？

  1010111+1=1011000，

  下一个组合为$\{x_6, x_4, x_3\}$

- 例2： $S=\{x_6, x_5 \ldots, x_1, x_0\}$的 哪个子集是子集列表中的第108个子集？

（注：列表上的位置是从 0 开始，第108个子集是指子集列表中对应108的子集）

108的二进制数：1101100

第108个子集为 $\{x_6, x_5, x_3, x_2\}$

字典序对应的组合生成

- 例：集合S={4, 3, 2, 1}的组合生成。

0000 → ∅		1000 → {4}	
0001 → {1}		1001 → {4,1}	
0010 → {2}		1010 → {4,2}	
0011 → {2,1}		1011 → {4,2,1}	
0100 → {3}		1100 → {4,3}	
0101 → {3,1}		1101 → {4,3,1}	
0110 → {3,2}		1110 → {4,3,2}	
0111 → {3,2,1}		1111 → {4,3,2,1}	

相邻组合可能相差较大

是否可以使得相邻的组合尽可能相似？

算法2：反射Gray码序生成算法

- 特点：相邻的组合**仅相差一个元素**

  （增加一个或者删除一个元素）

- 如：n（=1, 2, 3）元集的组合

$n=1$, $\varnothing$, $\{x_0\}$

0 1

$n=2$, $\varnothing$, $\{x_0\}$, $\{x_1, x_0\}$, $\{x_1\}$

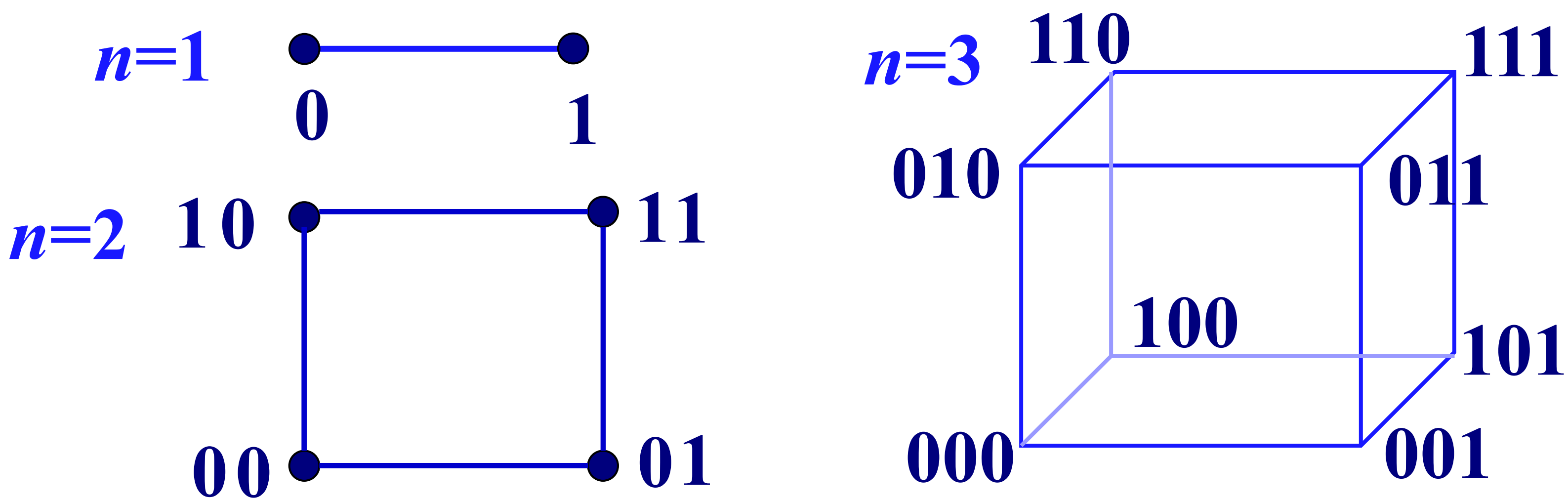
00 01 11 10

$n=3$

0	000	$\varnothing$
1	001	$\{x_0\}$
3	011	$\{x_0, x_1\}$
2	010	$\{x_1\}$
6	110	$\{x_2, x_1\}$
7	111	$\{x_2, x_1, x_0\}$
5	101	$\{x_2, x_0\}$
4	100	$\{x_2\}$

几何表示（Gray序）

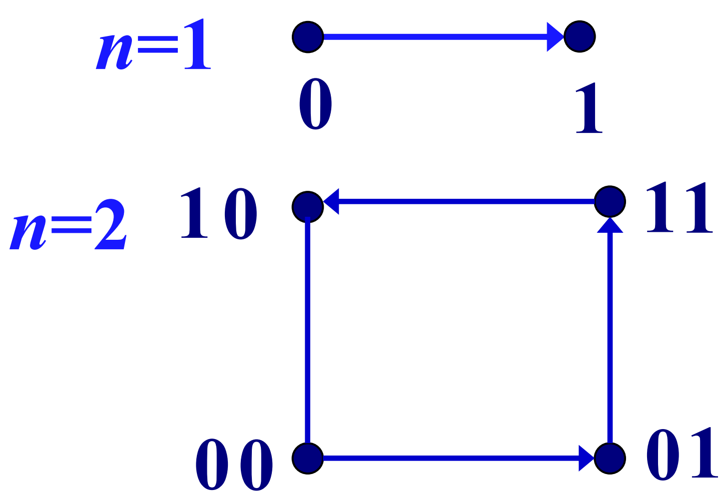
- n 元组看作是 n 维空间的点的坐标　（单位 n方体）
- 每两个点的坐标仅有一个位置不同时，有一条连线

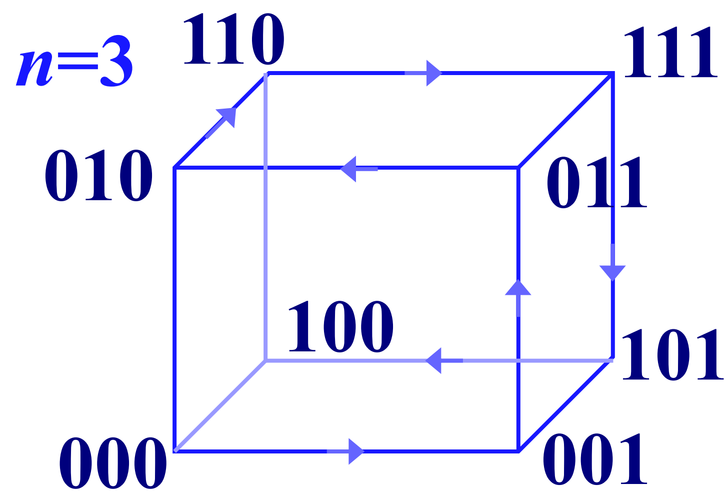


- 算法生成所有的 n元组：
  - ✓ 遍历 n 维空间的每个点，使得每个点与其后继只在一个位置不同；
  - ✓ 产生的路径称为n阶Gray码

几何表示（Gray序）

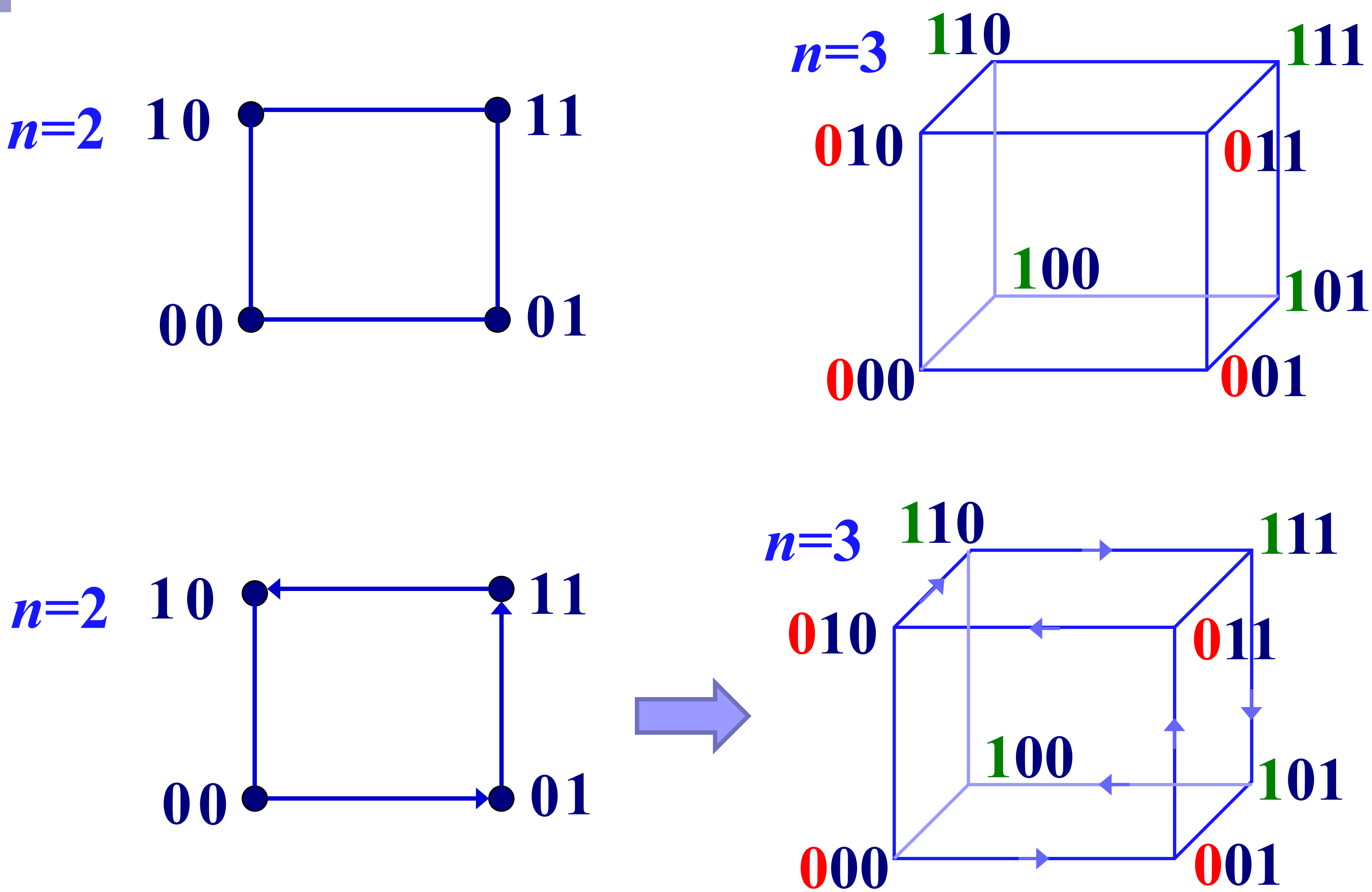
- n 元组看作是 n 维空间的点的坐标　（单位 n方体）
- 每两个点的坐标仅有一个位置不同时，有一条连线





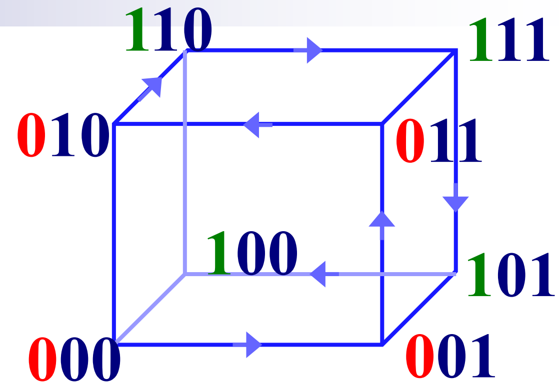
- 算法生成所有的 n元组：
  - ✓ 遍历 n 维空间的每个点，使得每个点与其后继只在一个位置不同；
  - ✓ 产生的路径称为n阶Gray码

遍历可以再经过一条边从终点返回到起点：循环Gray码



递归构造 *n* 阶Gray码（$n\geq 1$）：反射Gray码

n 阶Gray反射码的归纳定义



1. 1阶反射Gray码是 $\begin{matrix}0\\1\end{matrix}$ ；

2. 设n>1且n–1阶反射Gray码已经构造，如下构建 n 阶反射Gray码：

   (1) 以n–1阶反射Gray码所给出的顺序列出 0 和 1 的n–1元组，把 0 添到每个 n–1 元组的开头，

   (2) 再反序列出n–1阶反射Gray码的全部 n–1元组, 并把1加到全部 n–1元组的开头。

- n 阶反射Gray码以 00…0开始，并以 10…0结束。
- 因为00…0 与10…0只相差一位，因此该码是循环码。

1阶Gray码

0
1

2阶Gray码

0 0
0 1
1 1
1 0

反序

3阶Gray码

000
001
011
010
110
111
101
100

反序

- 相邻序数只有一位不同。
- 递归方法构造反射Gray码，生成组合。

问题：能否有直接的方法构造 n 阶反射Gray码？

以反射Gray码的顺序直接生成0, 1的n元组

1.初始：$a_{n-1}\ldots a_1 a_0 = 0\ldots00$

2. 当$a_{n-1}\ldots a_1 a_0 \neq 10\ldots0$ 时，进行以下操作：

(1) 计算$\sigma(a_{n-1}\ldots a_1 a_0) = a_{n-1}+\ldots+a_1+a_0$

(2) 如果$\sigma(a_{n-1}\ldots a_1 a_0)$是偶数，则改变 a_0（0变1或1变0）

(3) 否则，确定 j，使得$a_j = 1$且对于所有$i < j$, $a_i = 0$，然后，改变 a_{j+1}（0变1或1变0）.

称为逐次法。

每次改变均变化σ值的奇偶性

1 0 0 1 1 0 0 0 1 0 0 $\boxed{0}$

1 0 0 1 1 0 0 0 1 0 0 $\boxed{1}$

1 0 0 1 1 1 0 $\boxed{0}$ 1 0 0 0

1 0 0 1 1 1 0 $\boxed{1}$ 1 0 0 0

例：用逐次法生成4阶反射Gray码。

$\boxed{\sigma=0}$ $\boxed{\sigma=1}$ $\boxed{\sigma=2}$ $\boxed{\sigma=1}$ $\boxed{\sigma=2}$ $\boxed{\sigma=3}$

$0000 \rightarrow 0001 \rightarrow 0011 \rightarrow 0010 \rightarrow 0110 \rightarrow 0111$

$\rightarrow 0101 \rightarrow 0100 \rightarrow 1100 \rightarrow 1101 \rightarrow 1111 \rightarrow 1110$

$\rightarrow 1010 \rightarrow 1011 \rightarrow 1001 \rightarrow 1000$ 终止

$\boxed{\sigma=2}$

以反射Gray码的顺序直接生成0,1的n元组

1.初始：$a_{n-1}\ldots a_1a_0 = 0\ldots00$

2. 当$a_{n-1}\ldots a_1a_0 \neq 10\ldots0$ 时，进行以下操作：

(1) 计算$\sigma(a_{n-1}\ldots a_1a_0) = a_{n-1}+\ldots+a_1+a_0$

(2) 如果$\sigma(a_{n-1}\ldots a_1a_0)$是偶数，则改变 a_0（0变1或1变0）

(3) 否则，确定 j，使得$a_j = 1$且对于所有$i<j$, $a_i = 0$，

然后，改变a_{j+1}（0变1或1变0）.

称为逐次法。

每次改变均变化σ值的奇偶性

1 0 0 1 1 0 0 0 1 0 0 $\boxed{0}$

1 0 0 1 1 0 0 0 1 0 0 $\boxed{1}$

1 0 0 1 1 1 0 $\boxed{0}$ 1 0 0 0

1 0 0 1 1 1 0 $\boxed{1}$ 1 0 0 0

定理 4.3.1 对于每一个正整数n， 逐次法生成 n 阶反射Gray码

证明：对n进行归纳证明。

1. $n=1$ 时显然成立。
2. 假设对于$n-1$时，结论成立，即逐次法生成 $n-1$ 阶反射Gray码。
3. 当对于n时，证明逐次法生成 n 阶反射Gray码。

考虑 n 阶Gray码的前 2^{n-1} 个组合与后 2^{n-1} 个组合。

0 000
0 001
0 011
0 010
0 110
0 111
0 101
0 100
1 100
1 101
1 111
1 110
1 010
1 011
1 001
1 000

(1) 考虑 n 阶Gray码的前2^{n-1}个组合：

- ✓ 是由 n-1 阶Gray码在开头添加0形成，因此不会改变σ 值的奇偶性。
- ✓ 除第 2^{n-1}个元组（010…0）外，其余元组首位的 0不影响逐次法的应用，即逐次法用于前2^{n-1}–1个元组，与逐次法生成n-1阶Gray码的顺序一致。

由归纳假设，<u>逐次法可生成前一半的n阶Gray码</u>。

对 n 阶反射码的第2^{n-1}个元组 (010…0)，运用逐次算法：σ(010…0)＝1，则得到 (110…0) 正好是2^{n-1}+1个 n 阶反射Gray码。

0	000
0	001
0	011
0	010
0	110
0	111
0	101
0	100
1	100
1	101
1	111
1	110
1	010
1	011
1	001
1	000

(2)考虑 n 阶Gray码的后2^{n-1}个组合：

对任意前后连续的两个n元组：

$$1a_{n-2}\ldots a_1a_0 \to 1b_{n-2}\ldots b_1b_0,$$

只需证明逐次法确实从$1a_{n-2}\ldots a_1a_0$ 生成 $1b_{n-2}\ldots b_1b_0$。

在n-1阶反射Gray码中，有$a_{n-2}\ldots a_1a_0$在 $b_{n-2}\ldots b_1b_0$ 之后，即 $b_{n-2}\ldots b_1b_0 \to a_{n-2}\ldots a_1a_0$ 。

由于 $\sigma(a_{n-2}\ldots a_1a_0)$ 与 $\sigma(b_{n-2}\ldots b_1b_0)$ 奇偶性相反，

因此$\sigma(1a_{n-2}\ldots a_1a_0)$ 与 $\sigma(1b_{n-2}\ldots b_1b_0)$奇偶性也相反。

(a)若$\sigma(1a_{n-2}\ldots a_1a_0)$是偶数，那么$\sigma(a_{n-2}\ldots a_1a_0)$是奇数，

$\sigma(b_{n-2}\ldots b_1b_0)$是偶数，

$b_{n-2}\ldots b_1\ b_0$

$\Downarrow$

$a_{n-2}\ldots a_1\ a_0$

n-1阶

$a_i = b_i,\ i = 1, n\text{-}2$

$a_0 \neq b_0$

$1b_{n-2}\ldots b_1\ b_0$

$\Uparrow$

$1a_{n-2}\ldots a_1\ a_0$

n阶

0 000
0 001
0 011
0 010
0 110
0 111
0 101
0 100
1 100
1 101
1 111
1 110
1 010
1 011
1 001
1 000

反序

(2)考虑 n 阶Gray码的后2^{n-1}个组合：

对任意前后连续的两个n元组：

$$1a_{n-2}\ldots a_1a_0 \to 1b_{n-2}\ldots b_1b_0,$$

只需证明逐次法确实从$1a_{n-2}\ldots a_1a_0$ 生成 $1b_{n-2}\ldots b_1b_0$。

在n-1阶反射Gray码中，有$a_{n-2}\ldots a_1a_0$在 $b_{n-2}\ldots b_1b_0$ 之后，即 $b_{n-2}\ldots b_1b_0 \to a_{n-2}\ldots a_1a_0$ 。

由于 $\sigma(a_{n-2}\ldots a_1a_0)$ 与 $\sigma(b_{n-2}\ldots b_1b_0)$ 奇偶性相反，因此$\sigma(1a_{n-2}\ldots a_1a_0)$ 与 $\sigma(1b_{n-2}\ldots b_1b_0)$奇偶性也相反。

(a)若$\sigma(1a_{n-2}\ldots a_1a_0)$是偶数，那么$\sigma(a_{n-2}\ldots a_1a_0)$是奇数，

$\sigma(b_{n-2}\ldots b_1b_0)$是偶数，

由归纳假设， $a_{n-2}\ldots a_1a_0$由$b_{n-2}\ldots b_1b_0$改变b_0得到，因此，由$1a_{n-2}\ldots a_1a_0$改变a_0得到$1b_{n-2}\ldots b_1b_0$， 与逐次法一致。

0 000
0 001
0 011
0 010
0 110
0 111
0 101
0 100

1 100
1 101
1 111
1 110
1 010
1 011
1 001
1 000

反序

(2)考虑 n 阶Gray码的后2^{n-1}个组合：

对任意前后连续的两个n元组：

$$1a_{n-2}\ldots a_1a_0 \to 1b_{n-2}\ldots b_1b_0,$$

只需证明逐次法确实从$1a_{n-2}\ldots a_1a_0$ 生成 $1b_{n-2}\ldots b_1b_0$。

在n-1阶反射Gray码中，有$a_{n-2}\ldots a_1a_0$在 $b_{n-2}\ldots b_1b_0$ 之后，即 $b_{n-2}\ldots b_1b_0 \to a_{n-2}\ldots a_1a_0$ 。

由于 $\sigma(a_{n-2}\ldots a_1a_0)$ 与 $\sigma(b_{n-2}\ldots b_1b_0)$ 奇偶性相反，因此 $\sigma(1a_{n-2}\ldots a_1a_0)$ 与 $\sigma(1b_{n-2}\ldots b_1b_0)$奇偶性也相反。

(b) 若$\sigma(1a_{n-2}\ldots a_1a_0)$是奇数，那么$\sigma(a_{n-2}\ldots a_1a_0)$是偶数，$\sigma(b_{n-2}\ldots b_1b_0)$是奇数，

由归纳假设，$a_{n-2}\ldots a_1a_0$由$b_{n-2}\ldots b_1b_0$改变 b_{j+1}得到，其中 $b_j=1$，$0\le j< n-2$, 而对于所有$i<j$, $b_i=0$。

由于$\sigma(1a_{n-2}\ldots a_1a_0)$是奇数，因此，由$1a_{n-2}\ldots a_1a_0$改变$a_{j+1}$得到$1b_{n-2}\ldots b_1b_0$，与逐次法一致。由归纳法假设，结论成立。

0 000
0 001
0 011
0 010
0 110
0 111
0 101
0 100
1 100
1 101
1 111
1 110
1 010
1 011
1 001
1 000

反序

组合的两种生成方法

2^n 个二进制 n 元组的两种线性排序

- 从00…0 开始利用二进制算术的**字典序**
  - 与二进制数顺序一致
- 从00…0开始的**反射Gray码**
  - 相邻两个子集相差一个元素
  - 递归法、逐次法

问题：如何确定 n 元组在线性排序的准确位置？

■ 如何确定 n 元组在 Gray 码序表的准确位置？

给定Gray码$a_{n-1}\ldots a_1a_0$. 对于$i = 0, 1, \ldots, n\text{-}1$，设

$$b_i=\begin{cases}0, \text{若} a_{n-1}+\cdots+a_i \text{是偶数}\\ 1, \text{若} a_{n-1}+\cdots+a_i \text{是奇数}\end{cases},$$

此时，$a_{n-1}\ldots a_1a_0$在 Gray码序表的位置和$b_{n-1}\ldots b_1b_0$在字典序表上的位置相同。

即$a_{n-1}\ldots a_1 a_0$在Gray码序表的位置是

$$b_{n-1}\times 2^{n-1}+\ldots+b_1\times 2+b_0\times 2^0。$$

《计算机程序设计艺术》第4卷2册——生成所有元组和排列，Donald E. Knuth著，苏运霖译。

第四章：生成排列和组合

4.1 生成排列
4.2 排列中的逆序
4.3 生成组合
4.4 生成 r 子集

4.4 生成 *r*子集算法

例. 生成{4, 3, 2, 1}的所有2子集

方法一：

效率低！

1. 生成所有组合

0000	0001	0010	$\boxed{0011}$	0100	$\boxed{0101}$	$\boxed{0110}$	0111
1000	$\boxed{1001}$	$\boxed{1010}$	1011	$\boxed{1100}$	1101	1110	1111

2. 选出所有2子集

能否直接生成所有2子集？

{2, 1}, {3, 1}, {3, 2}, {4, 1}, {4, 2}, { 4, 3}

r子集的字典序：

- 令 $S=\{1, 2, \ldots, n\}$由前 n 个正整数组成。
  - 给出 S 的元素的一个自然顺序：

$$1 < 2 < \ldots < n$$

属于A或B，但不同时属于A和B

- 设A，B是 S 的两个 r 组合，若 $A \cup B \backslash A \cap B$ 中的最小整数属于A，则称 A先于B。

例 $\{1, 2, 3, 4, 5, 6, 7, 8\}$的两个5子集

$A=\{2，3，4，7，8\}$，$B=\{2，3，5，6，7\}$

$A \cup B \backslash A \cap B=\{4, 5, 6, 8\}$

A以字典序先于B

- 组合表示为子序列
  - 约定 $S=\{1, 2, \ldots, n\}$ 的 r 子集为如下形式：

$$a_1 a_2 \ldots a_r, \text{其中 } 1 \le a_1 < a_2 < \ldots < a_r \le n$$

例： $\{1, 2, 3, 4, 5, 6, 7, 8\}$的两个 5子集：

23478 先于 23567

若 $A \cup B \backslash A \cap B$ 中的最小整数属于A，则称 A先于B。

- 组合表示为子序列
  - 约定 $S=\{1, 2, \ldots, n\}$ 的 r 子集为如下形式：

$a_1 a_2 \ldots a_r$，其中 $1 \le a_1 < a_2 < \ldots < a_r \le n$

例: $S = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$的 5子集按字典序排序，则

第一个是: 12345 ，最后一个: 56789

12589的直接后继是 12678

- 以 1258 开头的后继：因为 12589 是最后一个，所以无后继
- 以 125 开头的后继：因为 12589 是最后一个，所以无后继
- 以 12 开头的后继：12678 为第一个
- 以 1 开头的后继：13456 为第一个
- 第 1 位比 1 大的后继：23456 为第一个

12467 的直接后继是12468，　　24679 的直接后继是24689

- 设 $S=\{1, 2,\ldots, n\}$, $a_1\ldots a_r$是 S 的一个 r子集。

  (1) 对任意的$1\leq i<j\leq r-1$,

  $a_1\ldots a_r$ 的以 $a_1\ldots a_j$开头的第一个后继一定先于以 $a_1\ldots a_i$开头的第一个后继（如果存在）。

例: $S=\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$的 5子集按字典序排序,

考虑 12467 的直接后续。

- 以 1246 开头的后继: 12468 是第一个
- 以 124 开头的后继： 12478 是第一个
- 以 12 开头的后继： 12567 是第一个
- 以 1 开头的后继： 13456 是第一个
- 第 1 位比 1 大的后继：23456 是第一个

- 设 $S=\{1, 2,\ldots, n\}$, $a_1\ldots a_r$是 S 的一个 r子集。

  (1) 对任意的$1\leq i<j\leq r-1$,

  $a_1\ldots a_r$ 的以 $a_1\ldots a_j$开头的第一个后继一定先于以 $a_1\ldots a_i$开头的第一个后继（如果存在）。

记以 $a_1\ldots a_j$ 开头的第一个后继：

$$a_1\ a_2\ldots a_i\ \ a_{i+1}\ \ldots\ a_j\ b_{j+1}\ldots\ b_r$$

以 $a_1\ldots a_i$ 开头的第一个后继：

$$a_1\ a_2\ \ldots\ a_i\ \ c_{i+1}\ \ldots\ c_r$$

一定有：$a_{i+1}<c_{i+1}<\ldots<c_r$，且 $a_{i+1}<a_{i+2}<\ldots\ a_j<b_{j+1}<\ldots<b_r$

令$A=\{a_1, a_2, \ldots, a_i,\ a_{i+1}, \ldots, a_j, b_{j+1}, \ldots, b_r\}$,

$B=\{a_1, a_2, \ldots , a_i, c_{i+1}, \ldots , c_r\}$,

则 a_{i+1}一定是$A\cup B\backslash A\cap B$中的最小数，因此(1)成立。

- 设 $S=\{1, 2, \ldots, n\}$, $a_1\ldots a_r$是 S 的一个 r子集。

(1) 对任意的$1\leq i<j \leq r-1$，

$a_1\ldots a_r$ 的以 $a_1\ldots a_j$ 开头的第一个后继一定先于以 $a_1\ldots a_i$ 开头的第一个后继（如果存在）。

(2) 如果$a_r< n$, 则 $a_1\ldots a_r$ 的直接后继为$a_1\ldots a_{r-1}\ a_r+1$.

以 $a_1\ldots a_{r-1}$ 开头的第一个后继

定理4.4.1 (1) 设 $a_1a_2\ldots a_r$ 是 $\{1, 2, \ldots, n\}$ 的 r 子集。在字典序中，第一个 r子集是 $12\ldots r$，最后一个 r子集是 $(n-r+1)(n-r+2)\ldots n$。

$a_1\ldots a_{r-1}$ 不是最后一个r子集

(2) 设$a_1a_2\ldots a_r \neq (n-r+1)(n-r+2)\ldots n$。令 k 是满足 $a_k < n$且使得a_k+1不同于$a_1a_2\ldots a_r$中任一数的最大整数。那么, 在字典序中, $a_1a_2\ldots a_r$ 的直接后继是

$$a_1a_2\ldots a_{k-1}(a_k+1)(a_k+2)\ldots(a_k+r-k+1).$$

例如：$n = 9$, $r=6$, $k=4$, $a_1a_2\ldots a_6$ 的直接后继是 $a_1a_2a_3(a_4+1)(a_4+2)(a_4+3)$.

定理4.4.1 (1) 设 $a_1a_2\ldots a_r$ 是 $\{1, 2, \ldots, n\}$ 的 r 子集。

在字典序中，第一个 r子集是 $12\ldots r$，最后一个 r子集是 $(n-r+1)(n-r+2)\ldots n$。

$a_1\ldots a_{r-1}$ 不是最后一个r子集

(2) 设$a_1a_2\ldots a_r \neq (n-r+1)(n-r+2)\ldots n$。令 k 是满足 $a_k < n$且使得a_k+1不同于$a_1a_2\ldots a_r$中任一数的最大整数。那么, 在字典序中, $a_1a_2\ldots a_r$ 的直接后继是

$$a_1a_2\ldots a_{k-1}\,(a_k+1)(a_k+2)\ldots(a_k+r-k+1).$$

直接后继求解算法:

1. 当$a_i<n$时$(1\leq i\leq r)$, 求a_i+1, 判断a_i+1是否属于$\{a_1, \ldots, a_r\}$;
2. 找出满足$a_i < n$, 且a_i+1不在$\{a_1,\ldots, a_r\}$中的最大的 i, 记为 k, 那么, 在字典序中, $a_1a_2\ldots a_r$ 的直接后继是

$$a_1a_2\ldots a_{k-1}\,(a_k+1)\,(a_k+2)\ldots(a_k+r-k+1)$$

1. 当$a_i<n$时$(1\leq i\leq r)$, 求a_i+1, 判断a_i+1是否属于$\{a_1, \ldots, a_r\}$;

2. 找出满足$a_i<n$, 且a_i+1不在$\{a_1,\ldots, a_r\}$中的最大的 i, 记为 k, 那么, 在字典序中, $a_1a_2\ldots a_r$ 的直接后继是

$$a_1a_2\ldots a_{k-1}\ (a_k+1)\ (a_k+2)\ldots(a_k+r-k+1)$$

例. 考虑{1, 2, 3, 4, 5, 6, 7, 8}的5子集:

$a_1a_2a_3a_4a_5$

5子集 1 3 4 5 7的直接后继是13458

a_i+1: 2 4 5 6 8, $k=5$

5子集 13578的直接后继是 13678

a_i+1: 2468, $k=3$

5子集 12458的直接后继是 12467

a_i+1: 2356, $k=4$

定理4.4.1 (1) 令$a_1a_2\ldots a_r$是$\{1, 2, \ldots, n\}$的一个 r 组合, 在字典序中, 第一个r子集是$12\ldots r$, 最后一个r 组合是 $(n-r+1)(n-r+2)\ldots n$。

(2) 设$a_1a_2\ldots a_r \neq (n-r+1)(n-r+2)\ldots n$。令 k 是满足 $a_k<n$且使得a_k+1不同于$a_1a_2\ldots a_r$中任一数的最大整数。

那么, 在字典序中, $a_1a_2\ldots a_r$的直接后继是

$$a_1a_2\ldots a_{k-1}(a_k+1)(a_k+2)\ldots(a_k+r-k+1).$$

证明: (1) 根据字典序的定义知, 第一个组合是$12\ldots r$, 最后一个是的$(n-r+1)(n-r+2)\ldots n$。

(2) 设 $a_1a_2\ldots a_r$不是最后一个 r子集, k 为使得$a_k<n$且a_k+1不同于$a_1a_2\ldots a_r$中任一数的最大整数。

下面分两种情况进行证明: $a_r<n$ 或 $a_r=n$

定理4.4.1 (1) 令$a_1a_2\ldots a_r$是$\{1, 2, \ldots, n\}$的一个 r 组合, 在字典序中, 第一个r子集是$12\ldots r$, 最后一个r 组合是 $(n-r+1)(n-r+2)\ldots n$。

(2) 设$a_1a_2\ldots a_r \neq (n-r+1)(n-r+2)\ldots n$。令 k 是满足 $a_k<n$且使得a_k+1不同于$a_1a_2\ldots a_r$中任一数的最大整数。

那么, 在字典序中, $a_1a_2\ldots a_r$的直接后继是

$$a_1a_2\ldots a_{k-1}(a_k+1)(a_k+2)\ldots(a_k+r-k+1).$$

证明： (a) $a_r<n$时， 显然 $a_1a_2\ldots a_r$的字典序直接后继是以$a_1a_2\ldots a_{r-1}$开头的第一个后继，即 $a_1\ldots a_{r-1}(a_r+1)$。

因为$a_r<n$时, a_r+1肯定不同于$a_1a_2\ldots a_r$中任一数，且r是满足条件的最大整数,

则由定理4.4.1生成的后继为$a_1\ldots a_{r-1}(a_r+1)$，与前面一致。

定理4.4.1(1) $a_1a_2\ldots a_r$是$\{1,2,\ldots,n\}$的一个r子集, 在字典序中, 第一个r子集是$12\ldots r$, 最后一个r子集是$(n-r+1)\ (n-r+2)\ldots n$。
(2)设 $a_1a_2\ldots a_r \neq (n-r+1)\ (n-r+2)\ldots n$。令 k 是满足 $a_k<n$ 且使得 a_k+1不同于$a_1a_2\ldots a_r$中任一数的最大整数。那么, 在字典序中, $a_1a_2\ldots a_r$ 的直接后继是 $a_1a_2\ldots a_{k-1}\ (a_k+1)\ldots(a_k+r-k+1)$.

证明：(b) 当$a_r=n$时， 假设有满足条件的k， 此时有 $k\neq r$。

共r-k-1项

a_1	a_2	…	a_k	a_{k+1}	a_{k+2}	a_{k+3}	…	a_{r-1}	$a_r=n$
a_1+1	a_2+1	…	a_k+1	$a_{k+1}+1$	$a_{k+2}+1$	$a_{k+3}+1$	…	$a_{r-1}+1$	$a_r=n$

共r-k-1项 $>a_{k+1}$

则必有 $a_{k+2}=a_{k+1}+1,\ a_{k+3}=a_{k+2}+1,\ \ldots,\ a_{r-1}=a_{r-2}+1,\ a_r=a_{r-1}+1=n$。

得$a_1\ldots a_r = a_1\ldots a_k\ ,\ a_{k+1},\ a_{k+1}+1,\ a_{k+1}+2,\ \ldots,\ a_{k+1}+(r-k-1)= a_r = n$

因此， $a_{k+1} = n-(r-k-1)$， $a_k+1< a_{k+1}$。

从而， $a_1\ldots a_r = a_1\ldots a_k\ ,\ n-(r-k-1),\ n-(r-k-2),\ldots,\ n$.

定理4.4.1(1) $a_1a_2\ldots a_r$是$\{1,2,\ldots,n\}$的一个r子集, 在字典序中, 第一个r子集是$12\ldots r$, 最后一个r子集是$(n-r+1)\ (n-r+2)\ldots n$。
(2)设 $a_1a_2\ldots a_r \neq (n-r+1)\ (n-r+2)\ldots n$。令 k 是满足 $a_k<n$ 且使得 a_k+1不同于$a_1a_2\ldots a_r$中任一数的最大整数。那么, 在字典序中, $a_1a_2\ldots a_r$ 的直接后继是 $a_1a_2\ldots a_{k-1}\ (a_k+1)\ldots(a_k+r-k+1)$.

证明：(b) 当$a_r=n$时，假设有满足条件的k，此时有 $k\neq r$。
得到 $a_1\ldots a_r = a_1\ldots a_k,\ a_{k+1},\ a_{k+1}+1,\ a_{k+1}+2,\ \ldots.,\ a_{k+1}+(r-k-1)$
$= a_1\ldots a_k,\ n-(r-k-1),\ n-(r-k-2),\ldots.,\ n.$

其中，$a_k+1< a_{k+1}=n-r+k+1$。
因此，$a_1a_2\ldots a_r$ 是以$a_1\ldots a_{k-1}a_k$ 开始的最后的 r 子集。
而$a_1\ldots a_{k-1}(a_k+1)(a_k+2)\ldots(a_k+r-k+1)$是以
$a_1\ldots a_{k-1}(a_k+1)$开始的第一个 r 子集，结论成立。

$\{1, 2,\ldots, n\}$的字典序 r 子集的生成算法

- 从$12\ldots r$ 开始，逐个列出直接后继，直至得到$(n-r+1)(n-r+2)\ldots n$

1. 初始： $a_1a_2\ldots a_r = 12\ldots r$;
2. 当$a_1a_2\ldots a_r \neq (n-r+1)(n-r+2)\ldots n$ 时，进行以下操作：
   (1) 确定最大整数k, 使得

   $$a_k+1 \leq n，且 a_k+1 \neq a_i\ (i=1,2,\ldots,r);$$

   (2) 用$a_1a_2\ldots a_{k\text{-}1}\ (a_k+1)\ldots(a_k+r-k+1)$替换 $a_1a_2\ldots a_r$.

例：应用算法生成{1,2,…,6}的所有4子集

a_i+1： 2345 2346 234 2356 235

1234 → 1235 → 1236 → 1245 → 1246

a_i+1： 236 2456

→1256 →1345 →1346→1356→1456

→2345→2346→2356→2456→ 3456

例： 生成{1, 2, 3, 4, 5, 6, 7}的所有5-元组

定理4.2.2 $\{1, 2, \ldots, n\}$的r子集 $a_1a_2\ldots a_r$ 出现在 $\{1, 2, \ldots, n\}$ 的 r 子集中的字典序中的位置号为：

$$\binom{n}{r}-\binom{n-a_1}{r}-\binom{n-a_2}{r-1}-\cdots-\binom{n-a_{r-1}}{2}-\binom{n-a_r}{1}$$

证明：位置号为所有r组合的个数减去其所有后继的个数。首先计算$a_1a_2\ldots a_r$ 的所有后继的个数。分两种情况：

(1) 当$1\le i\le r-1$时，设以$a_1a_2\ldots a_i$开头的 $a_1a_2\ldots a_r$的后继为 $a_1a_2\ldots a_i b_1\ldots b_{r-i}$，则有：

$$a_{i+1} < b_1 < \ldots < b_{r-i} \le n。$$

因此，以$a_1a_2\ldots a_i$开头的$a_1a_2\ldots a_r$的后继个数为：

$\binom{n-a_{i+1}}{r-i}$, $i=1,\ldots, r-1$

(2) 第一个位置比a_1大的后继的个数为$\binom{n-a_1}{r}$

因此，可得位置号。证毕。

定理4.2.2 $\{1, 2, \ldots, n\}$的r子集 $a_1a_2\ldots a_r$ 出现在 $\{1, 2, \ldots, n\}$ 的 r 子集中的字典序中的位置号为：

$$\binom{n}{r}-\binom{n-a_1}{r}-\binom{n-a_2}{r-1}-\cdots-\binom{n-a_{r-1}}{2}-\binom{n-a_r}{1}$$

例. 求{ 1, 2, …, 8 }的 4子集 1258 的字典序位置。

解：1258的位置是：

$$\binom{8}{4}-\binom{7}{4}-\binom{6}{3}-\binom{3}{2}-\binom{0}{1}=22$$

第四章 内容小节

- 排列生成算法：
  - 递归生成算法
  - 邻位对换算法
  - 从逆序生成排列
- 组合生成算法
  - 字典序
  - 反射Gray码
  - 基于字典序的 r子集生成算法

《计算机程序设计艺术》第4卷2册——生成所有元组和排列，Donald E. Knuth著， 苏运霖译。

Donald E. Knuth 高德纳

- **1938年1月10日生于美国威斯康星州密尔沃基市**
- **他的超凡智力在8岁时就显示出来了，用“Ziegler’s Giant Bar”里面的字母，写单词。裁判准备了一份2500个单词的列表，而高德纳却写出了4500多个单 词，获得了冠军。他的赛后感言是“我还能写出更多”。**
- **1960年Donald E. Knuth 22岁毕业，由于“成绩过于优异”，同时被授予学士和硕士学位。**
- **他在36岁的时候就获得了图灵奖（Unix的发明人之一Ken Thompson 是到40多岁才拿图灵奖的)**
- **Knuth总共教了28个博士生。不知道怎么搞的，他觉得28这个数字很好，于是就决定再也 不带博士生了。**

- **1962年，世界上一流的出版社Addison-Wesley艾迪生-韦斯利出版社约初露头角的高德纳写一本编译器和程序设计方面的书，这件原本寻常的事最终成就了计算机科学史上的一个奇观。**
- **1962年约的稿，高德纳一直写到1966年还没交(写了4年），编辑找到高德纳，说这都四年了你写了多少啊，高德纳说，才写3000页手稿。编辑大囧，忙问都3000页了你怎不交，高德纳答曰，急啥， 我还没写到正题呢。编辑彻底雷住了，说那你出个多卷本吧……**

《计算机程序设计艺术》，就这么诞生了，计划写7卷。

- 1968年，《计算机程序设计艺术》(TAOCP)的第一卷：基本算法正式出版了。
- 微软首席执行官比尔盖茨在1995年接受一次采访时说，“如果你认为你是一名真正优秀的程序员，就去读第一卷，确定可以解决其中所有的问题。”盖茨本人读这本书时用去了几个月的时间，并同时进行了难以置信的训练。
- 盖茨还说：“如果你能读懂整套书的话，请给我发一份你的简历。”高德纳本人的说法更犀利：要是看不懂，就别当 程序员。
- 1970年第二卷半数值算法出版，1973年第三卷排序与查找出版，这三卷书立即被计算机界惊为神作，在那几年就卖出去100多万套，至今 仍是编程书籍中的最高经典。

- **1974年，高纳德获得图灵奖，保持着获奖年龄最小的纪录**

BIRTH:

January 10, 1938, in Milwaukee, Wisconsin.

EDUCATION:

Graduated from Milwaukee Lutheran High School (1956); BS in mathematics from the

DONALD ("DON") ERVIN KNUTH

United States – 1974

CITATION

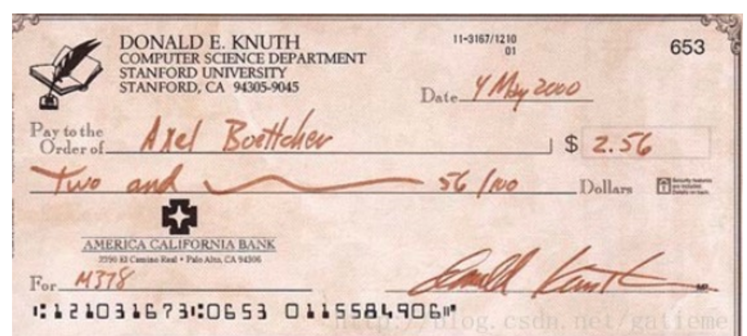
For his major contributions to the analysis of algorithms and the design of programming languages, and in particular for his contributions to the "art of computer programming" through his well-known books in a continuous series by this title.

- 封笔十年，创造了三个重要的成果：
  - 排版系统TEX
  - 字体设计系统METAFONT
  - 文学化编程(Literate Programming)

谁发现TEX的一个错误，就付他2.56 美元，第二个错误5.12美元，第三个10.24美元……以此类推。

另 一个奖项是找出其著作中错误的人能得到2.56美元，因为“256美分刚好是十六进制的一美元”

有网友戏说，什么是聪明：在 Knuth 的书中找到错误；什么是愚蠢：去兑现那张两块五毛六的支票。



- **2008年，在TAOCP的前三卷面市30年之后，第四卷终于面世了**

*正如当年**Linux**的作者**Linus**说：上帝在梦中告诉我，我做出了最优秀的操作系统。高德纳回答说：我可没这么说过*