

这是 Google 对 <https://blog.csdn.net/wbin233/article/details/72998375> 的缓存。 这是该网页在 2021年5月13日 07:28:04 GMT 的快照。 [当前页](#)在此期间可能已经更改。 [了解详情](#).

[完整版本](#) [纯文字版本](#) [查看源代码](#)

提示: 要在此页面上快速找到您的搜索字词，请按 **Ctrl+F** 或者 **⌘+F** (Mac) ， 然后使用查找栏搜索。

康托展开和逆康托展开

原创

wbin233

2017-06-10 17:56:16

13402

收藏 39

版权

分类专栏: [算法](#) [acm](#) 文章标签: [算法](#) [康托展开](#) [逆康托展开](#) [acm](#)

文章目录

- [简述](#)
- [原理](#)
- [康托展开](#)
- [逆康托展开](#)
- [示例:](#)
- [应用](#)

简述

康托展开是一个全排列到一个自然数的双射，常用于构建hash表时的空间压缩。设有n个数（1，2，3，4,...,n），可以有组成不同(n!种)的排列组合，康托展开表示的就是在n个不同元素的全排列中, 比当前排列组合小的个数，那么也可以表示当前排列组合在n个不同元素的全排列中的名次（当前的名次 = 比当前排列组合小的个数 + 1）。

原理

$$X=a[n]*(n-1)!+a[n-1]*(n-2)!+...+a[i]*(i-1)!+...+a[1]*0!$$

其中, a[i]为整数，并且0 <= a[i] <= i, 0 <= i < n, 表示当前未出现的的元素中排第几个，这就是康托展开。

例如有3个数（1，2，3），则其排列组合及其相应的康托展开值如下：

排列组合	名次	康托展开	值
123	1	$0 * 2! + 0 * 1! + 0 * 0!$	0
132	2	$0 * 2! + 1 * 1! + 0 * 0!$	1
213	3	$1 * 2! + 0 * 1! + 0 * 0!$	2
231	4	$1 * 2! + 1 * 1! + 0 * 0!$	3
312	5	$2 * 2! + 0 * 1! + 0 * 0!$	4
321	6	$2 * 2! + 1 * 1! + 0 * 0!$	5

比如其中的 231：


- 想要计算排在它前面的排列组合数目（123，132，213），则可以转化为计算比首位小即小于2的所有排列「1 * 2!」，首位相等为2并且第二位小于3的所有排列「1 * 1!」，前两位相等为23并且第三位小于1的所有排列（0 * 0!）的和即可，康托展开为：1 * 2! + 1 * 1+0 * 0=3。
- 所以小于231的组合有3个，所以231的名次是4。


康托展开

再举个例子说明。

在（1，2，3，4，5）5个数的排列组合中，计算 34152的康托展开值。

- 首位是3，则小于3的数有两个，为1和2，a[5]=2，则首位小于3的所有排列组合为 a[5]*(5-1)!
- 第二位是4，则小于4的数有两个，为1和2，注意这里3并不能算，因为3已经在第一位，所以其实计算的是在第二位之后小于4的个数。因此a[4]=2
- 第三位是1，则在其之后小于1的数有0个，所以a[3]=0
- 第四位是5，则在其之后小于5的数有1个，为2，所以a[2]=1
- 最后一位就不用计算啦，因为它在它之后已经没有数了，所以a[1]固定为0





举报

- 根据公式:

$$X = 2 * 4! + 2 * 3! + 0 * 2! + 1 * 1! + 0 * 0! = 2 * 24 + 2 * 6 + 1 = 61$$

所以比 34152 小的组合有61个，即34152是排第62。

具体代码实现如下：（假设排列数小于10个）

```
static const int FAC[] = {1, 1, 2, 6, 24, 120, 720, 5040, 40320, 362880}; // 阶乘
int cantor(int *a, int n)
{
    int x = 0;
    for (int i = 0; i < n; ++i) {
        int smaller = 0; // 在当前位之后小于其的个数
        for (int j = i + 1; j < n; ++j) {
            if (a[j] < a[i])
                smaller++;
        }
        x += FAC[n - i - 1] * smaller; // 康托展开累加
    }
    return x; // 康托展开值
}
```

tips: 这里主要为了讲解康托展开的思路，实现的算法复杂度为 $O(n^2)$ ，实际当 n 很大时，内层循环计算在当前位之后小于当前位的个数可以用 **线段树**来处理计算，而不用每次都遍历，这样复杂度可以降为 $O(n \log n)$ 。

逆康托展开

一开始已经提过了，康托展开是一个全排列到一个自然数的**双射**，因此是可逆的。即对于上述例子，在 (1, 2, 3, 4, 5) 给出61可以算出起排列组合为 34152。由上述的计算过程可以容易的逆推回来，具体过程如下：

- 用 $61 / 4! = 2$ 余 13，说明 $a[5]=2$,说明比首位小的数有2个，所以首位为3。
- 用 $13 / 3! = 2$ 余 1，说明 $a[4]=2$ ，说明在第二位之后小于第二位的数有2个，所以第二位为4。
- 用 $1 / 2! = 0$ 余 1，说明 $a[3]=0$ ，说明在第三位之后没有小于第三位的数，所以第三位为1。
- 用 $1 / 1! = 1$ 余 0，说明 $a[2]=1$ ，说明在第二位之后小于第四位的数有1个，所以第四位为5。
- 最后一位自然就是剩下的数2啦。
- 通过以上分析，所求排列组合为 34152。

具体代码实现如下：（假设排列数小于10个）

```
static const int FAC[] = {1, 1, 2, 6, 24, 120, 720, 5040, 40320, 362880}; // 阶乘

//康托展开逆运算
void decantor(int x, int n)
{
    vector<int> rest; // 存放当前可选数，保证有序
    for(int i=1;i<=n;i++)
        v.push_back(i);

    vector<int> ans; // 所求排列组合
    for(int i=n;i>=1;i--)
    {
        int r = x % FAC[i-1];
        int t = x / FAC[i-1];
        x = r;
        a.push_back(v[t]); // 剩余数里第t+1个数为当前位
        v.erase(v.begin()+t); // 移除选做当前位的数
    }
}
```

示例：

60. 第k个排列

难度 中等 327 收藏 分享 切换为英文 关注 反馈

给出集合 $[1, 2, 3, \dots, n]$ ，其所有元素共有 $n!$ 种排列。

按大小顺序列出所有排列情况，并一一标记，当 $n = 3$ 时，所有排列如下：

```
1. "123"
2. "132"
3. "213"
4. "231"
5. "312"
6. "321"
```

给定 n 和 k ，返回第 k 个排列。

说明：

- 给定 n 的范围是 $[1, 9]$ 。
- 给定 k 的范围是 $[1, n!]$ 。

示例 1:

输入: $n = 3, k = 3$
输出: "213"

示例 2:

输入: $n = 4, k = 9$
输出: "2314"

<https://blog.csdn.net/wbin233>

直接套逆康托展开即可，需要处理的是求的是第k个排列，那么其对应的康托展开值应该减1（ $k - 1$ ）。

```
class Solution {
public:
    // 逆康托展开
    string getPermutation(int n, int k) {
        static constexpr int FAC[] = {1, 1, 2, 6, 24, 120, 720, 5040, 40320, 362880}; // 阶乘

        vector<int> rest; // 存放当前可选数，有序
        for(int i = 1; i <= n; i++)
            rest.push_back(i);

        k--; // 要先 -1 才是其康托展开的值
        string ans = "";
        ans.reserve(n);

        for(int i = n; i >= 1; i--)
        {
            int r = k % FAC[i-1];
            int t = k / FAC[i-1];
            k = r;
            ans += to_string(rest[t]); // 剩余数里第t+1个数为当前位
            rest.erase(rest.begin() + t); // 移除选做当前位的数
        }

        return ans;
    }
};
```

应用

应用最多的场景也是上述讲的它的特性。

- 给定一个自然数集合组合一个全排列，所其中的一个排列组合在全排列中从小到大排第几位。
在上述例子中，在（1，2，3，4，5）的全排列中，34152的排列组合排在第62位。
- 反过来，就是逆康托展开，求在一个全排列中，从小到大的第n个全排列是多少。
比如求在（1，2，3，4，5）的全排列中，第62个排列组合是34152。[注意具体计算中，要先-1才是其康托展开的值。]
- 另外康托展开也是一个数组到一个数的映射，因此也是可用于hash，用于空间压缩。比如在保存一个序列，我们可能需要开一个数组，如果能够把它映射成一个自然数，则只需要保存一个整数，大大压缩空间。比如八数码问题。

点赞30 评论13 分享 收藏39 打赏 举报 关注 一键三连

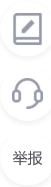
都能看懂的康托展开

lrbless的博客 6726

康托展开简介：官方简介：康托展开是一个全排列到一个自然数的双射，常用于构建哈希表时的空间压缩。康托展...

康托展开

zhongkeli的专栏 2万+



康托展开 康托展开的公式是 $X = a_n \cdot (n-1)! + a_{n-1} \cdot (n-2)! + \dots + a_i \cdot (i-1)! + \dots + a_2 \cdot 1! + a_1 \cdot 0!$ 其中, a_i 为当前未出现的元素...

优质评论可以帮助作者获得更高权重 评论

相关推荐

康托展开和逆康托展开 Geek 1724
简述 康托展开是一个全排列到一个自然数的双射, 常用于构建hash表时的空间压缩。设有n个数 (1, 2, 3, 4,...,n)

算法基础: 康托展开与逆康托展开 ajaxlt的博客 7388
康托展开与逆康托展开 序言: 本文记录康托展开与逆康托展开的原理以及其应用。1.概述 举例而言, 对于 1 ~ 4 的一...

康托展开与实例 qq_49662108的博客 828
康托展开与实例 康托展开是一个全排列到一个自然数的双射, 类似于哈希, 每一种排列都有一个唯一的对应字典序...

康托展开详解 -csdn博客 i-Curve的博客 217
康托展开详解 -csdn博客 定义: 康托展开是一个全排列到一个自然数的双射, 常用于构建哈希表时的空间压缩。康...

康托展开 obsorb_knowledge的博客 244
康托展开是什么? 定义: $X = a_n \cdot (n-1)! + a_{n-1} \cdot (n-2)! + \dots + a_i \cdot (i-1)! + \dots + a_2 \cdot 1! + a_1 \cdot 0!$ a_i 为整数, 并且 $0 \leq a_i < i$ 简单点说就是, 判...

康托展开与逆康托展开详解 zyz_bz的博客 387
文章目录康托展开运用板子 逆康托展开板子 康托展开 康托展开是一个全排列到一个自然数的双射, 常用于构建哈希...

康托展开的代码 10-08
实现康托展开hash的代码, pascal语言通过将数组转变为数字

【算法】康托展开和逆康托展开 立志如山, 行道如水 337
文章目录康托展开 逆康托展开 康托展开 康托展开是一个全排列到一个自然数的双射, 常用于构建hash表时的空间压...

康托展开 (维基百科) MrBlank的ACM记事本 514
via: <https://zh.wikipedia.org/wiki/康托展开> 康托展开是一个全排列到一个自然数的双射, 常用于构建哈希表时的空间...

康托展开及其逆运算 详解 继续激情, 继续奋斗。 1万+
康托展开 康托展开逆运算 详解

全排列计算 (康托展开) 52learn 1万+
题目描述 给出一个1~n的全排列中的某一个, 求它是按字典序排列的第几个。输入输出格式 输入格式: 第一行, 一...

详细讲解康托展开集齐基础例题+模板 塞满箱子的过程 264
单纯讲讲怎么用怎么算, 说些人话。预备概念: 排列: 对于一个数字n的排列就是含有[1,n]所有数字的序列。也就...

康托展开及其应用 liulingbo918 1015
康托展开是一种特殊的哈希函数, 用阶乘的线性组合来表示一个数x, 即 $x = a[n] \cdot n! + a[n-1] \cdot (n-1)! + \dots + a[1] \cdot 1! + \dots$

CSDN开发者助手, 常用网站自动整合, 多种工具一键调用
CSDN开发者助手由CSDN官方开发, 集成一键呼出搜索、万能快捷工具、个性化新标签页和官方免广告四大功能。...

简要介绍康托展开 蚩尤煜的博客 2187
本篇文章并不会详细讲解康托展开的数学原理, 有兴趣的朋友可以另行寻找。康托展开 作用: 判断这个数在其各个...

关于康托展开的用途及写法 sluqy671的专栏 989
在处理八数码这一类需要用到全排列的问题的时候, 存储往往是一个难题, 因为明明只有n!种情况, 数字的长度却...

c语言之康托展开 潘猫猫的世界 1044
/* $X = a_n \cdot (n-1)! + a_{n-1} \cdot (n-2)! + \dots + a_i \cdot (i-1)! + \dots + a_1 \cdot 1! + a_0 \cdot 0!$ 其中, a_i 为整数, 并且 $X = a_n \cdot (n-1)! + a_{n-1} \cdot (n-2)! + \dots + a_i \cdot (i-1)! + \dots$

康托展开及逆康托展开 落花时节又逢君 88
概念: 康托展开是一个全排列到一个自然数的双射, 常用于构建哈希表时的空间压缩。康托展开的实质是计算当前...

©2020 CSDN 皮肤主题: 大白 设计师: CSDN官方博客 返回首页

关于我们 招贤纳士 广告服务 开发助手 400-660-0108 kefu@csdn.net 在线客服 工作时间 8:30-22:00
公安备案号11010502030143 京ICP备19004658号 京网文[2020] 1039-165号 经营性网站备案信息 北京互联网违法和不良信息举报中心
网络110报警服务 中国互联网举报中心 家长监护 Chrome商店下载 ©1999-2021北京创新乐知网络技术有限公司 版权与免责声明 版权申诉
出版物许可证 营业执照



wbin233
码龄5年 暂无认证

28	13万+	9万+	8万+	
原创	周排名	总排名	访问	等级
1085	34	82	48	101
积分	粉丝	获赞	评论	收藏



私信

关注



举报

搜博主文章

热门文章

LIS算法: 最长上升子序列 15910

使用Flask-Mail和qq邮箱SMTP服务发送邮件 14473

康托展开和逆康托展开 13394

Android 从相册中选择照片并返回 6853

python 列表去重（不可变类型和可变类型） 4600

分类专栏

 android 3篇

 acm 6篇

 java 6篇

 python 4篇

 杂七杂八

 算法 4篇

最新评论

java8 ArrayList源码阅读【2】 - 总结
Tisfy: 此贴甚妙!

分布式一致性协议: Basic Paxos原理及...
清浅浅浅g: 写得真不错, 可以配合《从 Paxos 到 ZooKeeper》食用, 讲述的逻辑有 ...

康托展开和逆康托展开
独恒而后: 好的, 我知道了, 谢谢博主, 但是你的代码确实有点问题 (我觉得 (只 ...

康托展开和逆康托展开
wbin233: 0.0 只能说使用场景有限 (对n而言) 咯。求第几个排列你用next_perm ...

康托展开和逆康托展开
独恒而后: 还有你发出的代码没有一份是对的.....

最新文章

分布式一致性协议: Basic Paxos原理及推导

glusterfs IO缓存池

gluster安装完全指南

2018年 3篇

2017年 16篇

2016年 6篇

2015年 3篇

目录