

基础套路 I



SELF-INTRO

阮行止，湖南师大附中/哈尔滨工业大学

洛谷网校讲师

优敏思oi教研组



INTRO

这篇课件用于讲述一些基础套路。

前置技能：NOIP提高组知识

约定几个英文缩写：

e.g.	例如	动物都是生物，e.g. 猫是生物。
etc.	等等	动物中有猫，狗，etc.
P.S.	备注	P.S. 这篇课件是rxz做的。

剪枝

一个很朴素的思想：在搜索过程中，如果预知某条路一定不可能到达答案，则放弃这条路。

之所以叫做“剪枝”，是因为它剪掉了搜索树上的一些子树。

埃及分数

<https://www.luogu.org/problemnew/show/UVA12558>

ID

迭代加深搜索：DFS与BFS的结合。

BFS：保证找出最浅层的解，但是费空间

DFS：省空间，但是解未必是最浅层的

ID：限制搜索层数，然后**DFS**；如果找不到答案，则放宽层数，继续**DFS**。

埃及分数

这题可以使用**ID.**
注意剪枝即可。

二分

首先提一下二分的感性理解。
(看黑板)

三分

三分用于求单峰函数（或单谷函数）的极值。
看黑板。

学分绩问题

哈工大的学分绩是这样算的：

$$\text{平均学分绩 } G = \frac{\sum \text{学分} \cdot \text{成绩}}{\sum \text{学分}}$$

给定每门课的学分 w 和你对这门课的预估成绩 c 。

你可以缓考至多 k 门课，被缓考的课暂时不计入学分绩。

求可能达到的最高学分绩。

<http://acm.hit.edu.cn/contest/170/problem/A>

子问题

子问题: $k=1$?

子问题: $k=n-1$?

现在 k 取任意数, 怎么办?

01 分数规划

现在考虑二分答案。

需要给出一种方式，来判断能否达到 p 的学分绩。

推柿子

现在把实际问题转化成数学问题：

确定数组 b ， b_i 为0则表示缓考了这门课；否则表示正常计算。

最大化

$$G = \frac{\sum r_i \cdot b_i}{\sum c_i \cdot b_i}$$

其中 $r_i = c_i \cdot w_i$

推柿子

$$G = \frac{\sum r_i b_i}{\sum c_i b_i}$$

于是 $G \cdot \sum c_i b_i = \sum r_i b_i$

现在给定 p ，我们要判断答案是否比 p 大。也就是判断：

是否存在一组 b ，使得

$$\sum r_i b_i = G \cdot \sum c_i b_i > p \cdot \sum c_i b_i$$

亦即

$$\sum r_i b_i - p \cdot \sum c_i b_i > 0$$

推柿子

现在问题就转化成：判断能否有数组 b 满足

$$\sum r_i b_i - p \cdot \sum c_i b_i > 0$$

如果有，那么还可以继续寻找更优的答案；

如果没有，则我们对 p 的估计过高了。

推柿子

$$\sum r_i b_i - p \cdot \sum c_i b_i = \sum b_i (r_i - p \cdot c_i)$$

对于每个 i ，既然给定了 p ，那么 $(r_i - p \cdot c_i)$ 就是个定值。

既然是定值，那么我们问题转换为：

给定数组 a ，剔除掉一些数，问能否使数组的和大于0。

直接排序就完事了。

代码

那我现场写一个？

区间数颜色

给定一个长度为 n 的序列，每个元素有颜色。应对 m 个询问：

Ask(l, r): 询问 $[l, r]$ 区间内有多少种颜色。

允许离线。

暴力 A

暴力 A: 直接跑一遍询问区间去统计。

单次询问复杂度 $O(n)$, 取决于数据, 无法改变。

总复杂度 $O(n^2)$

复杂度完全取决于数据, 我们无法改进。

暴力 B

暴力B：维护数组 w ， $w[i]$ 表示 i 这个颜色出现了多少次。

记 cnt 为当前处理的 $[l, r]$ 这个区间的颜色数。

现在如果我们手上已经有了一个区间的信息，如何求出下一个区间的信息？

移动 l, r 指针，使之走到新的询问。每次移动指针的时候就更新信息。 l, r 走到之后， cnt 即为答案。

暴力 B

这个暴力算法的复杂度取决于什么？

取决于我们需要移动多少次指针。

我们需要移动多少次指针，和我们处理询问的顺序有关！

比如，三个询问 $[1, 2]$, $[10, 10000]$, $[5, 6]$

明显 $a \rightarrow c \rightarrow b$ 这个处理顺序，移动指针的次数小于 $a \rightarrow b \rightarrow c$ 。

莫队算法

国际上叫做MO's Algorithm.

提出者是**2010**年集训队莫涛（长郡中学）。

基本思路：改变这些询问的顺序，使之对我们有利。

P.S. 学术上的名称大概是 “Query square root decomposition”.

莫队算法

考虑所有询问。

先分 \sqrt{n} 个桶，把询问按照 l 扔进 $\frac{l}{\sqrt{n}}$ 这个桶。

然后，针对每个桶：将其中的询问按照 r 排序。

这套事情做完之后，直接跑暴力B。

复杂度： $O(m\sqrt{n})$ 。

复杂度分析

为什么复杂度就可以 $O(m\sqrt{n})$?

考虑 l 指针:

每个桶内的元素, l 相差不会超过 \sqrt{n} , 故 l 指针在每次询问的时候至多移动 \sqrt{n} 次。故 l 指针在整个程序中移动 $O(m\sqrt{n})$ 次。

考虑 r 指针:

每个桶内, r 都是有序的, 所以在每个桶内 r 最多移动 n 次。

一共有 \sqrt{n} 个桶, 故 r 移动 $O(n\sqrt{n})$ 次。

代码实现

写两个函数：`add`, `del`用于跳指针。

排序：不需要显式地执行分块操作，只需要在排序的时候，以`l`所在的块作为第一关键字，以`r`作为第二关键字。

其它题目

区间询问出现次数多于**3**的颜色的个数。

区间询问出现次数多于**k**的颜色的个数。

区间询问众数。

[AHOI2013] 作业

<https://www.luogu.org/problemnew/show/P4396>

莫队算法的适用范围

允许离线；

可以写出复杂度较好的`add`和`del`函数；

没有修改操作。

带修莫队

<http://ruanx.pw/bzojch/p/2120>

带修莫队

按 **x** 所在块为第一关键字；**y** 所在块为第二关键字；**t** 为第三关键字来排序。其他操作一样。

这里要注意，块大小应该改成 $n^{2/3}$ 。

总复杂度 $O(n^{5/2})$ 。

单调栈

一群人排队，问每个人最近的比他高的人是谁。

要求 $O(n)$ 。

单调队列

给定序列 a 和一个 k ，问所有长度为 k 的子区间的最大值分别是多少。

要求 $O(n)$ 。

子序列

给定 s 和一个长度为 n 的正整数序列。求出最短的子区间长度，满足这个子区间的和不小于 s 。

e.g. $n=5, s=11, a=(1, 2, 3, 4, 5)$ 。

$(3, 4, 5)$ 的和为12，不小于11且最短。

P.S. 题目来源：POJ3061 (SUBSEQUENCE)

子序列

如何暴力？

枚举左右端点。 $O(n^2)$.

如何优化？

二分长度。 $O(n \log n)$.

子序列

这题有 $O(n)$ 做法。

我们取两个指针，最开始都放在 a_1 位置。

这两个指针会夹住区间 $a_{[l,r]}$ 。我们现在可以通过移动这两个指针，来表示任意一段区间。

指针移动的时候，区间和的变化可以快速得到。

子序列

操作完了之后，现在我们把 l 固定在 a_1 ，要找到一个最短的区间，使得区间和 $\geq s$ 。

把右指针不停地往右移，直到 $a_{[l,r]}$ 的和 $\geq s$ 。

现在的 r 就是左端点为 a_1 时的最佳答案。

那么，如何算出其他左端点的最佳答案？

子序列

我们把 l 指针往右移一位，然后再去考虑移动 r 。

由于 l 指针右移了一位，区间里少了一个数，所以区间和变小了。

因此， r 指针要么不移，要么只能往右移。我们继续往右移动 r 指针，直到新的区间和 $\geq s$ 为止。

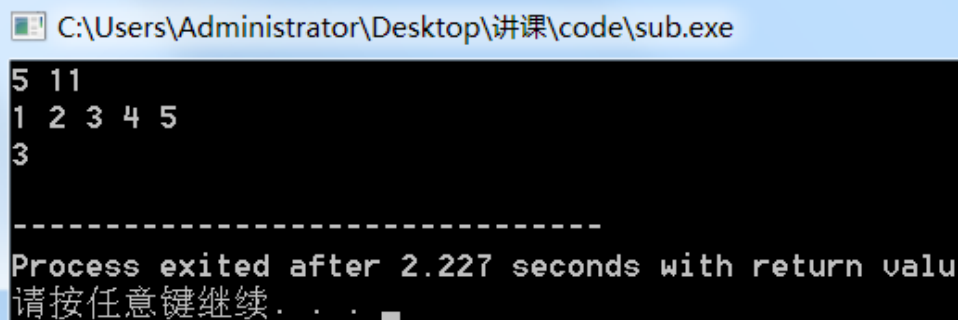
子序列

```
scanf("%d%d",&n,&s);
for(i=1;i<=n;i++)
    scanf("%d",&a[i]);

l=1,r=1,su=a[1];
while(l<=n)
{
    while(su<s && r<=n)
        r++,su+=a[r];

    if(su>=s) ans=min(ans,r-l+1);
    su-=a[l];
    l++;
}

printf("%d\n",ans);
```



```
C:\Users\Administrator\Desktop\讲课\code\sub.exe
5 11
1 2 3 4 5
3

-----
Process exited after 2.227 seconds with return value
请按任意键继续...
```

子序列

为什么我们要这样干呢？

因为，左指针和右指针都只会向右移动。

再怎么动也只能走过整个序列的长度！ $O(n)$ 。

陵墓设计

给定 n ，问将 n 拆成连续的数的平方和的方案。

输出这些方案。 $n \leq 10^{14}$ 。

$$\text{e.g. } 2030 = 21^2 + 22^2 + 23^2 + 24^2 = 25^2 + 26^2 + 27^2.$$

陵墓设计

既然 $n \leq 10^{14}$, 我们只需要考虑 $\leq 10^7$ 的数的平方。

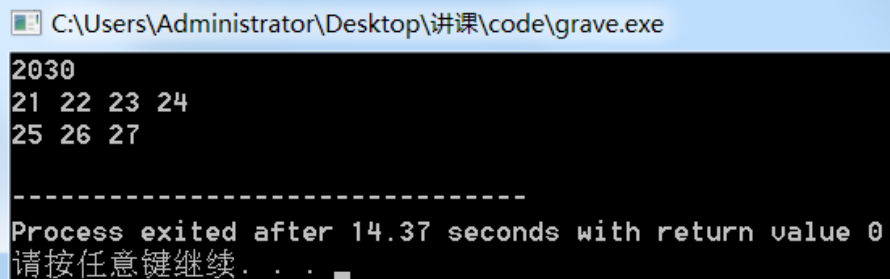
算出数组 a , 其中 $a[i] = i^2$.

现在我们需要在 a 上面取子区间, 使得区间和恰为 n .

把上面的代码稍微修改一下就行了。

陵墓设计

```
n=1000000;  
scanf("%d",&s);  
for(i=1;i<=n;i++)  
    a[i]=i*i;  
  
l=1,r=1,su=a[1];  
while(l<=n)  
{  
    while(su<s && r<=n)  
        r++,su+=a[r];  
  
    if(su==s)  
    {  
        for(i=l;i<=r;i++)  
            printf("%d ",i);  
        puts("");  
    }  
    su-=a[l];  
    l++;  
}
```



```
C:\Users\Administrator\Desktop\讲课\code\grave.exe  
2030  
21 22 23 24  
25 26 27  
-----  
Process exited after 14.37 seconds with return value 0  
请按任意键继续. . .
```

阅读材料

此文总结了一系列two-pointers算法。

作者forever97.

<https://www.cnblogs.com/forever97/category/909925.html>

前缀和

给定一个序列 a (初值全为 0)。有很多次询问，每个询问形如：

A l r 询问将 $a_{[l,r]}$ 的区间和。

每次询问的复杂度要求 $O(1)$ 。

前缀和

我们搞出一个数组 s , 其中 $s_i = a_1 + a_2 + \cdots + a_i$.

那么: $a_l + a_{l+1} + \cdots + a_r = s_r - s_{l-1}$.

完事!

区间加

给定一个序列 a (初值全为 0)。有很多次操作，每个操作形如：

A l r k 将 $a_{[l,r]}$ 每个值加上 k 。

最后输出整个数组。复杂度要求 $O(n)$ 。

区间加

代码：（已经预先指定 $a[0]=0$ ）

```
1. void add(int l,int r,int k)
2. {p[l]+=k,p[r+1]-=k;}
3.
4. void get_a()
5. {
6.     for(int i=1;i<=n;i++)
7.         a[i]=a[i-1]+p[i];
8. }
```

区间加

我们很自然地想到：

如果我们知道每一个元素比前一个元素大多少，我们显然可以推出整个序列。

e.g. 已知 $a_1 = 2$ 。 a_2 比 a_1 大3， a_3 比 a_2 小4。

那么可以推出： $a_2 = a_1 + 3 = 5$ ， $a_3 = a_2 - 4 = 1$ 。

区间加

区间加 $[1, r]$ ，实际上是发生了这两件事：

$a[1]$ 比前一个元素多了 k ；

$a[r+1]$ 比前一个元素少了 k 。

麻烦自己脑补一下：)



区间加

我们用数组 \mathbf{p} 表示刚刚的差值， $\mathbf{p}[i] = \mathbf{a}[i] - \mathbf{a}[i-1]$ 。

那么：区间加 $[l, r]$ ，可以化为这两个操作：

$\mathbf{p}[l] += k;$

$\mathbf{p}[r+1] -= k;$

因此，一次区间加只修改这两个元素；

最后利用 \mathbf{p} 数组求出 \mathbf{a} 数组，即为答案。

差分与前缀和

通过上述的两个方法，我们能轻易地处理这两类问题：

- 数组固定，然后大量询问；
- 大量做区间加，最后要你给出这个数组。

完形填空

现在你要做一篇完形填空。你的策略是：对于每个空的四个选项，选出在本文中出現过的单词。如果有多个，则选择离这个空最近的那个单词。

全文长度100w字节。

子树统计

给定一棵树。要求支持两种操作：

- 修改 将点 x 的权值改为 p
- 查询 查询 x 的子树的权值和

离散化

考虑莫队的那道例题。

如果序列长度只有**10w**，颜色值域是**1e8**，怎么办？

明显顶多只有**10w**种颜色有用。我们统计一下出现过的所有颜色，然后给编号就行了。

可以直接排序。如果懒，也可以用**set**。

二维数点

给定平面直角坐标系上的一大堆整点。

问：对于每个点，有多少个点 x, y 坐标均小于这个点。

分块

区间数颜色，强制在线。

分块

区间众数。强制在线。

HASH冲突

<https://www.luogu.org/problemnew/show/P3396>

论文题

给定无向图 $G(V, E)$ ，每个点要么是白色要么是蓝色。

最开始所有的点都是蓝色。有两种操作：

- **Turn x** 将x点颜色翻转
- **Ask x** 询问与x相连的点中，有几个蓝色点

CDQ分治

CDQ（陈丹琦，雅礼中学）分治是一种思想：

把所有的事件进行分治。

考虑前面的事件对后面事件的影响。

典型例子：归并排序求逆序对。

二维数点

再来看二维数点。**CDQ**分治能怎么做？

直接按 x 进行一遍排序。

然后开始分治。分治过程和按 y 归并排序一样；但是在合并信息的时候，我们考虑前半部分的点对后半部分统计答案的影响。

由于左边的点 x 值必然比右边的小（已经按 x 排过一遍序了！）

所以只需要看 y 。此时左右两边 y 分别是有序的，很容易统计。

三维数点

<https://www.luogu.org/problemnew/show/P3810>

BITSET神教

对于每个点 p ，处理出在第 k 维比它小的点的集合： S_{pk}

那么，这个点的答案是：

$$\text{count}(S_{p1} \cap S_{p2} \cap \cdots \cap S_{pm})$$

由于**bitset**常数很小，这题时间上没什么问题。

空间是 $O(n^2)$ 的，吃不消。采用分块：每一维只维护 \sqrt{n} 个点的具体信息，查询其它点的时候，从离它最近的重点开始，暴力求出这个点的信息。

时间复杂度挺暴力的.....大概勉强卡过。

GYM 100342J

给一个邻接矩阵，求有多少条路径可以由**A**出发，经过**B**，再经过**C**，最后回到**A**。

这里 $n \leq 1500$ 。

对于每个点，找到出、入的边，放进两个**bitset**。

枚举所有中转点对**(B,C)**，统计有多少个**A**能 $A \rightarrow B \rightarrow C \rightarrow A$ 。

这种**A**的个数，就是： $\text{count}(\text{In}_B \cap \text{Out}_C)$ 。勉强能卡过。

传递闭包

<https://www.luogu.org/problemnew/show/P4306>

传递闭包

先拓扑排序成**DAG**。

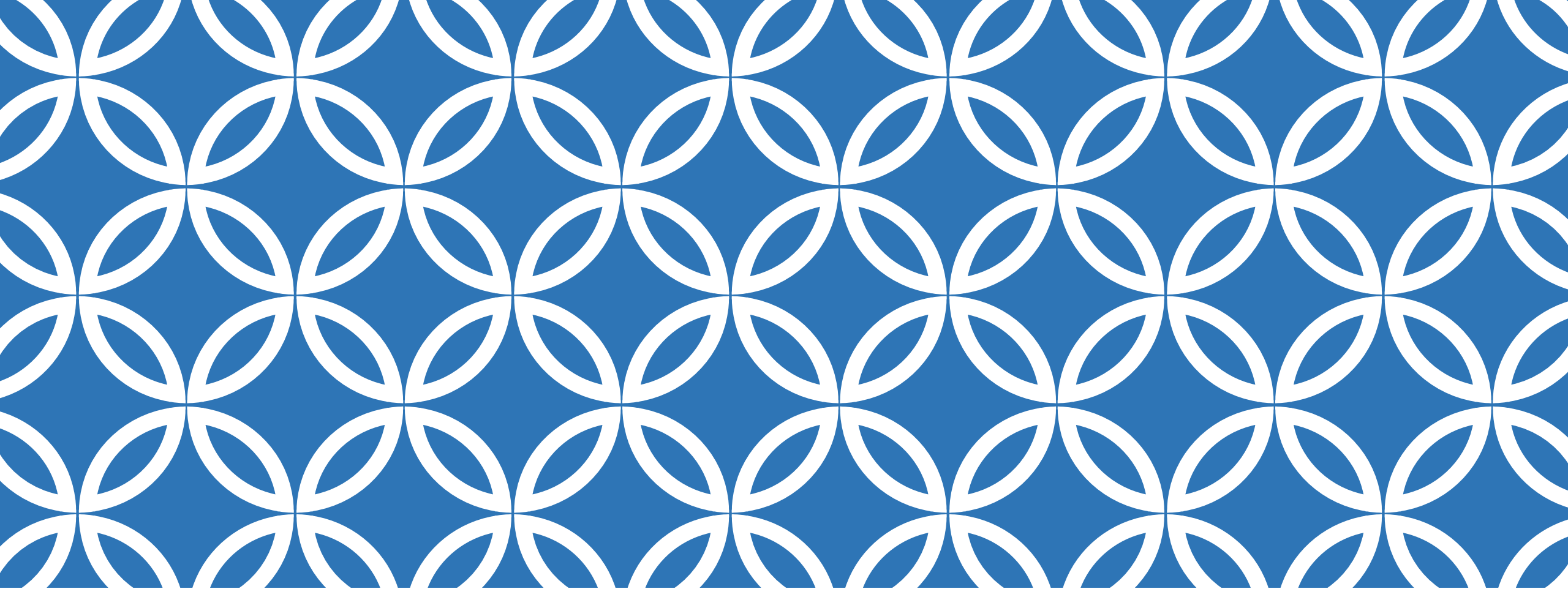
用**bitset** S_A 表示**A**能抵达的点。那么

$$S_A = S_{p1} \cap S_{p2} \cap \cdots \cap S_{pk}$$

其中 p_i 为**A**可以直接抵达的点。

对于所有的点，搞出**S**就能统计答案了。复杂度 $O\left(\frac{n^2}{32}\right)$

事实上，就算不缩点，直接以**Floyd**的方式来做，仍然能过.....



END

rxz@luogu.org