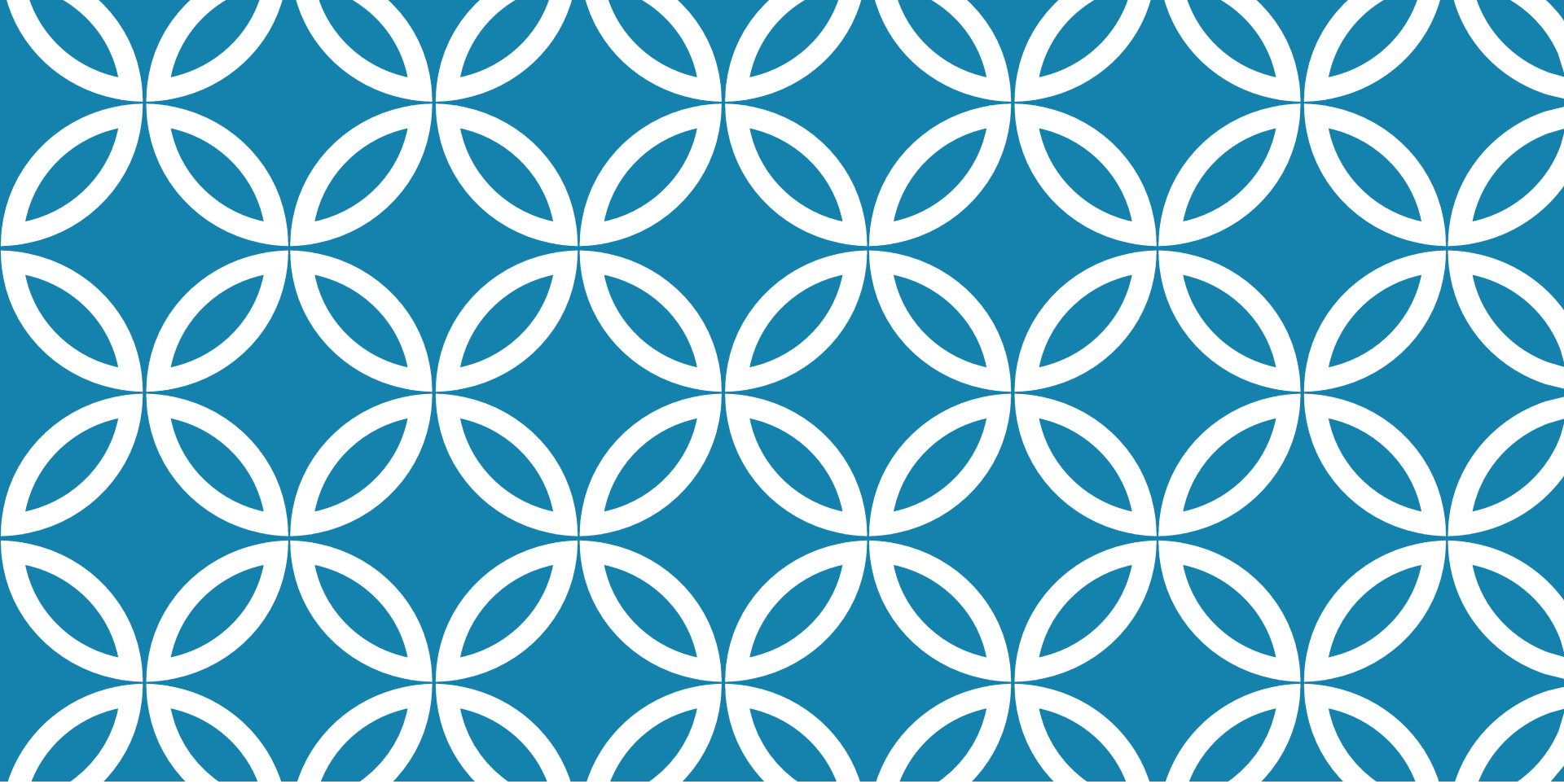


构造

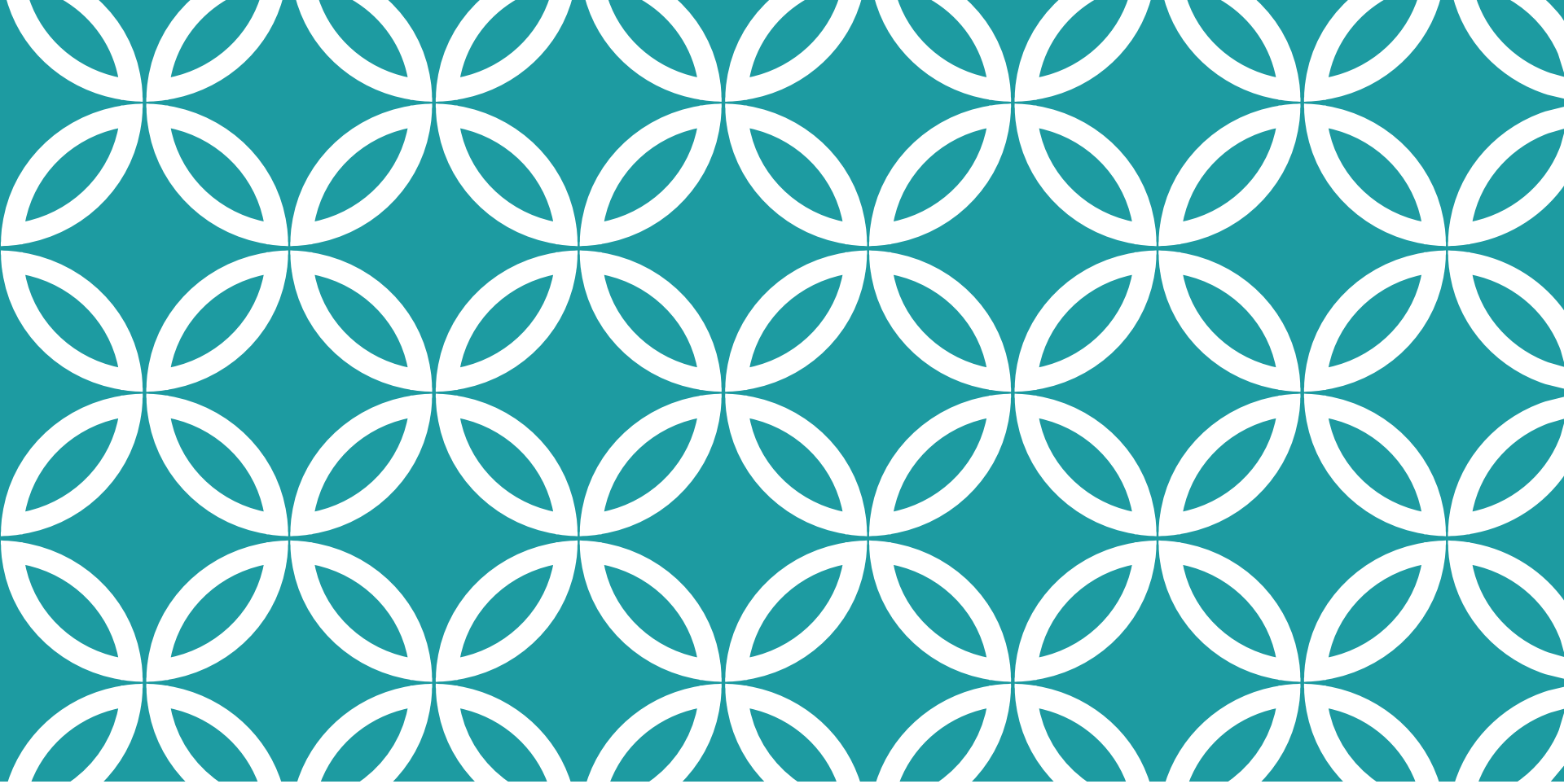
北京大学
沈 洋

作诠释

- 构造是**优美**的
- 构造是**有趣**的
- 构造是**困难**的
- 构造需要**知识**
- 构造需要**构思**
- 构造需要**创造**
- shenmegui!

作诠释

- 解法往往不唯一
- 甚至解也不唯一
- 往往有一个巧妙的idea
- 往往易于实现
- ……



构造解

利用题目条件
分析性质
直接构造

Divisible Subset

Time Limit: 1s

- 给定一个含n个整数的multiset
- 找出一个非空子集，满足子集中元素的和能被n整除
- 或判定这样的集合不存在

- $n \leq 10^5$

Divisible Subset

Time Limit: 1s

- 背包?
- 好像不太行的样子

- 加强版
- 找出一个非空**区间**，满足**区间**中元素的和能被n整除

Divisible Subset

Time Limit: 1s

- 区间和
- 前缀和作差
- 问题转化：是否有两个不同的前缀和$(\mathrm{mod}\ n)$

- $n+1$个前缀和
- n种不同的取值
- 抽屉原理
- 必有解

Divisible Subset

Time Limit: 1s

- 搞笑：如果改成$\mathrm{mod}\ (n+1)$还一定有解么？

Hack It!

Time Limit: 1s

- 令$f(x)$表示x在十进制下各位数字之和
- 给定一整数a 构造l, r
- 使得$\sum_{i=l}^{r} f(i) \equiv 0 \ (mod \ a)$

- $1 \leq a \leq 10^{18}$
- $1 \leq l, r \leq 10^{200}$

Hack It!

Time Limit: 1s

- 子任务：给定l, r，求$\sum_{i=l}^{r} f(i) \pmod a$
- 简单计算每一位上每个数字出现次数即可

- 考虑 x 和 $10^m + x$ 两个数$(x < 10^m)$
- $f(x)$与$f(10^m + x)$相差1
- 当$[l, r]$由$[1,10^m]$变为$[x + 1,10^m + x]$时 $\sum_{i=l}^{r} f(i)$增加x
- $[x + 1,10^m + x] | x = a - \left(\sum_{i=1}^{10^m} f(i) \bmod a\right)$满足条件

A Problem Concerning LCS

Time Limit: 1s

- 给定一个仅由A, C, T, G组成的长度为n的字符串a
- 要求构造一个仅由A, C, T, G组成的长度为n的字符串b
- 使得a和b的最长公共子序列最短

- $n \leq 10^6$

A Problem Concerning LCS

Time Limit: 1s

- 提供此题的人表示常数比较大的$O(n)$算法和常数较小的$O(n\log n)$算法均能通过此题
- 后缀树/后缀自动机/后缀数组?

A Problem Concerning LCS

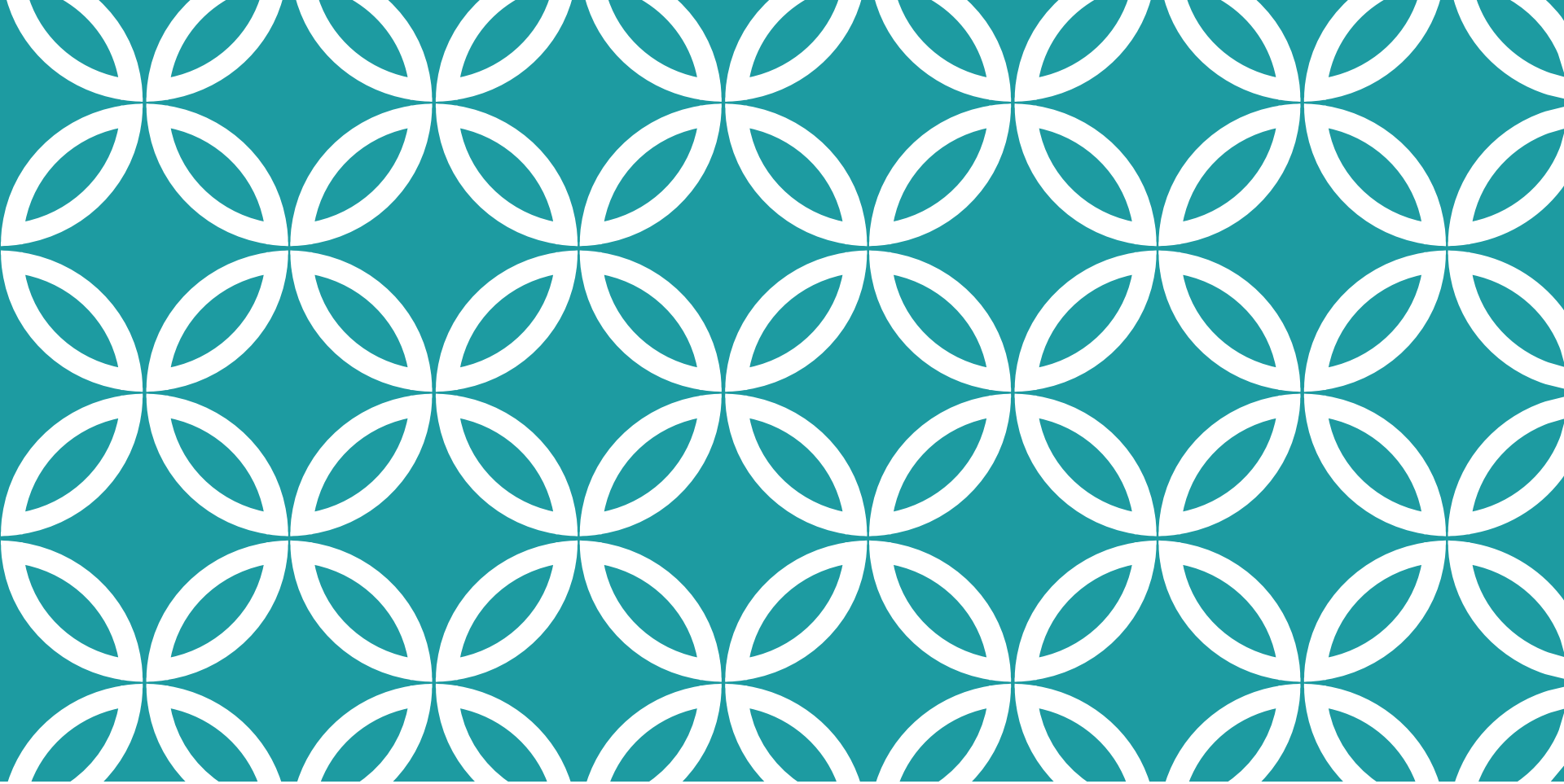
Time Limit: 1s

- 大胆猜测
- 由出现次数最少的字母组成的字符串

A Problem Concerning LCS

Time Limit: 1s

- 小心求证
- 不妨假设出现次数最少的字母是A，且出现了x次
- 显然$x \leq \frac{n}{4}$
- 以全A来构造串b，则LCS长度为x
- 假设存在串b'，与a的LCS小于x
- 则b'中每个字母出现次数均小于x
- 则b'长度小于n，出现矛盾



构造解

现有算法
稍加修改的现有算法

图的还原

Time Limit: 1s

- 已知一个n个顶点的无重边无自环的无向图的各点度数
- （请允许我叫它三无图）
- 构造一个可能的原图
- 或判断数据有误

- $n \leq 10^3$

图的还原

Time Limit: 1s

- 经典贪心算法
- 首先初始化一张空图
- 每个点的剩余度数表示这个点还需要连几个点
- 每次选择剩余度数最大的点，记其度数为x
- 连接该点与其他点中剩余度数最大的x个点
- 若与其他点中剩余度数非零的点不足x个则无解
- 否则更新剩余度数，重复上述过程

图的还原

Time Limit: 1s

- 思考：如果只需要判定能否构造呢?
- 上述构造算法可在$O(n\log n)$实现判定
- Erdős–Gallai Theorem
- $d_1, d_2 \ldots d_n (d_1 \geq d_2 \geq \cdots \geq d_n)$是一个合法度数序列的充要条件是$\sum d_i$为偶数且对于$k = 1 \ldots n$，下式成立

$$\sum_{i=1}^{k} d_i \leq k(k-1) + \sum_{i=k+1}^{n} \min(k, d_i)$$

- $O(n)$判定

修改边权

Time Limit: 1s

- 给定一个n个顶点，m条边的有向图
- 其中某些边权可能为负，保证没有负环
- 修改一些边的边权，使得：

a) 新图中每一条路径的长度都非负

b) 对于任意两个顶点u, v，u到v的任意两条路径都满足新图中两条路径的长度差等于原图中的长度差。

- 构造修改方案
- $n \leq 2 \cdot 10^4, m \leq 2 \cdot 10^5$

修改边权

Time Limit: 1s

- 怎么样的修改满足条件b?
- 设某一种修改已经满足了条件b
- 设点1到i的路径长度增量为d_i
- 那么对于某条原来某条长度为l的边$i \to j$
- 新长度必须为$l - d_i + d_j$
- 另一方面，任意一组d_i都对应一个满足条件b的修改
- 满足条件b的修改：确定$\{d_i\}$，把$i \to j$的长度$-d_i + d_j$

修改边权

Time Limit: 1s

- 如何满足a条件?
- $l - d_i + d_j \geq 0$
- $d_i - d_j \leq -l$
- 差分约束系统
- Bellman-Ford算法求解

修改边权

Time Limit: 1s

- 作用
- 消除负权 但不改变最短路

- 应用
- Johnson算法

Bags and Coins

Time Limit: 2.5s

- 有s个硬币，n个包
- 一个包可以放在其他包里面，可以多层嵌套
- 如果拿出某个硬币必须要打开第i个包我们就说这个硬币在第i个包里
- 现已知第i个包里总共有a_i个硬币
- 构造一种满足上述条件的方案
- 或确定问题无解
- $1 \leq n, s, a_i \leq 70000$

Bags and Coins

Time Limit: 2.5s

- 如何确定是否有解?
- 问题转化
- 选一些a_i（其中必须包括最大的那个，若有多个最大包括一个即可），使得他们的和等于s，求方案
- 背包+bitset
- 前$i-1$个物品时可以达到的总和的集合为S，加入第i个物品之后变为$S \mid (S \ll a_i)$

Bags and Coins

Time Limit: 2.5s

- 如何构造方案?
- 要重构方案，我们只需知道达到某个体积时是从哪个体积加入了哪个物品即可
- 有多种可能只需保留一个——不妨保留第一个
- 若$x \in S \mid (S \ll a_i)$但$x \notin S$说明第一次达到x体积
- 此时记$from[x] = i$
- 利用bitset的与和异或计算$S \mid (S \ll a_i)$与S的差
- 枚举bitset元素

存不下

Time Limit: 5s

- 有如下算法将一个n位01串X转成一个$2n$位01串：

1. 给定参数M, C
2. 令H = Mod(1371,2^M), s = 0
3. 依次对原串每位X_i执行：

```
s = RotL((343291835 3*s+C) mod 2^32,32)*461845907 mod 2^32
H = H xor (s mod 2^M) xor X_i
H = RotL(H,M)
H = (H*5+3864292196) mod 2^M
```

输出H的二进制位从低到高的第M位、第M-1位

存不下

Time Limit: 5s

- 现给定n, M, C
- 要求输出串中1最少
- 构造输入串

- $1 \leq n \leq 5000,\ 3 \leq M \leq 20,\ 0 \leq C < 2^{31}$
- $n \cdot 2^M \leq 5 \cdot 10^7$
- 内存限制 8 MB

存不下

Time Limit: 5s

- 记第i次执行完后的s, H分别为s_i, H_i
- s_i是与输入串x无关的某种常数
- H_i是x前i位的某种Hash
- H_i只与H_{i-1}和x_i有关
- 考虑依次枚举x的每一位
- 如果两种枚举的H_i相等，它们对于后续的枚举是等价的
- 以$f[i][H_i]$为状态DP

存不下

Time Limit: 5s

- 如何构造方案?
- 记录$f[i][H_i]$转移自上层的哪个H_{i-1}即可
- 存不下!
- 压缩记录数据量

存不下

Time Limit: 5s

- 有损压缩——时间换空间
- 无法记录转移路径的每一步
- 但可以记录转移路径上特定位置的H_i
- 分治/分块

存不下

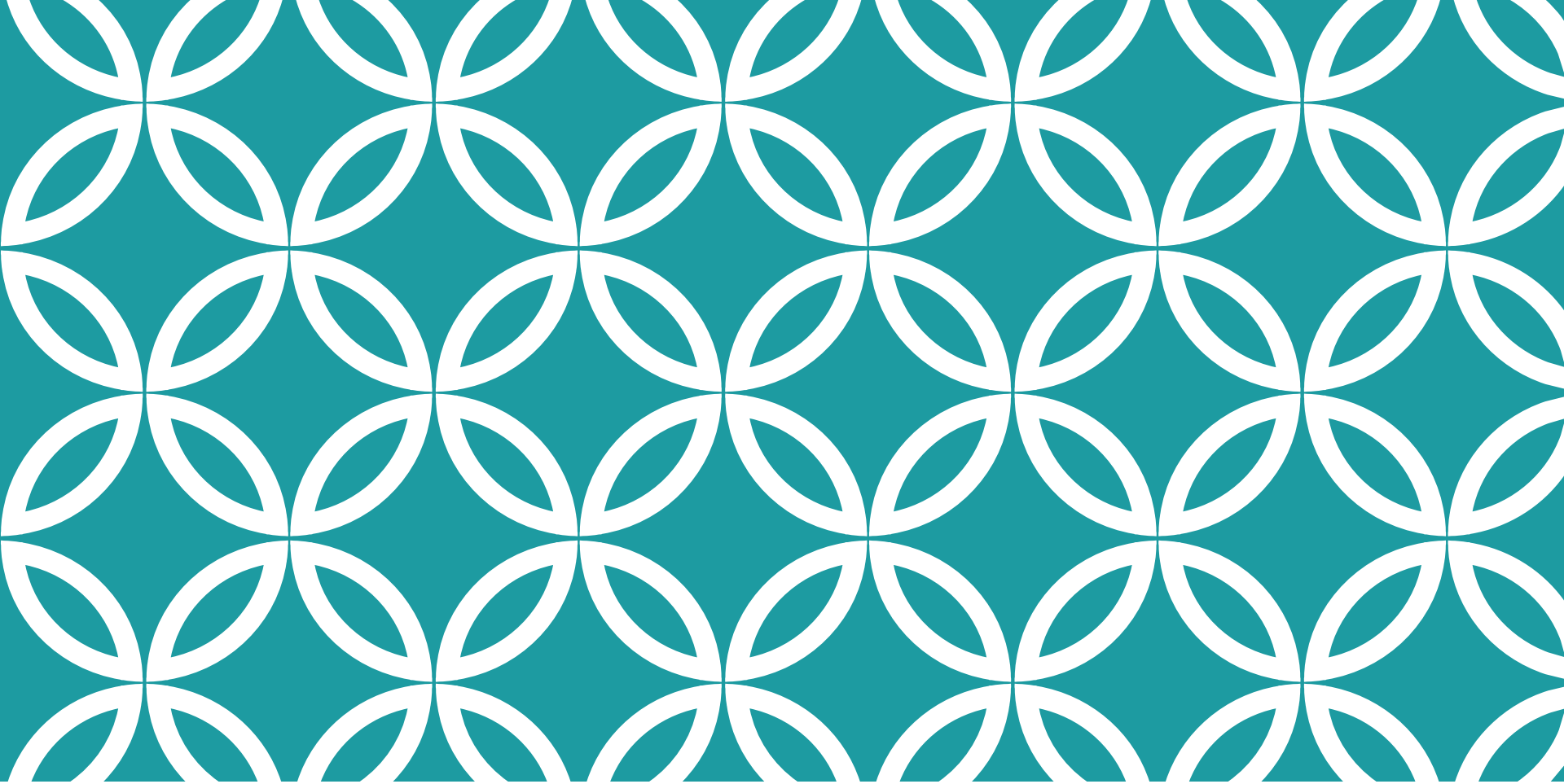
Time Limit: 5s

- 无损压缩
- 只记录添加了0/1而不记录具体的H_{i-1}是否可行？
- 本题似乎可以：通过H_i, x_i可反推H_{i-1}
- 仍然存不下
- 本题的特殊性
- H_i的0转移和1转移分别等价于$H_i \text{ xor } 1$的1转移和0转移
- 减少一半记录量（记录的信息需要稍加修改）

存不下

Time Limit: 5s

- 思考：如果通过H_i, x_i不可以反推H_{i-1}？
- 一个节点入边可能很多，但出边只有两条
- 反向DP



构造策略

限制可用操作
构造一个程序
交互

Stack Machine Programmer

Time Limit: 1s

- Stack Machine是一种抽象计算机
- 只能处理整数
- 只有一个栈作为存储器
- 支持以下指令：

Stack Machine Programmer

Time Limit: 1s

指令	作用
NUM X	将非负整数X压入栈中
POP	弹出栈顶的数
INV	将栈顶的数变为其相反数
DUP	将栈顶的数复制一份并压入栈中
SWP	交换栈顶的两个数
ADD	弹出栈顶两个数并将他们的和压栈
SUB	弹出栈顶两个数并将他们的差压栈
MUL	弹出栈顶两个数并将他们的乘积压栈
DIV	弹出栈顶两个数并将他们商的整数部分压栈
MOD	弹出栈顶两个数并将他们相除的余数压栈

Stack Machine Programmer

Time Limit: 1s

- 二元运算以先弹出的数作为右操作数，后弹出的数作为左操作数
- 一个程序是指一个顺序执行的指令序列
- 输入：开始运行时栈中唯一的数
- 输出：程序运行结束时栈中唯一的数
- 构造一个能满足给出的n个要求的程序
- 每个要求形如(V, R)，表示对于输入V，你的程序的输出应为R

Stack Machine Programmer

Time Limit: 1s

- $n \leq 5, 0 \leq V \leq 10, 0 \leq R \leq 20$
- 你构造的程序长度不能超过10^5

Stack Machine Programmer

Time Limit: 1s

- 提供了加减乘除取模运算
- 插值
- 本质上是构造了一系列函数$f_i(x)$
- 当$x = V_i$时$f_i(x) = 1$
- 当$x = V_j, j \neq i$时$f_i(x) = 0$
- 答案为$\sum_{i=1}^{n} R_i \cdot f_i(x)$

Stack Machine Programmer

Time Limit: 1s

- 构造通用的程序实现判断输入与已知常数是否相等
- 可以实现判断两个二进制位是否相等
- 可以实现将一个整数拆成二进制
- 可以通过二进制判断两数是否相等

Stack Machine Programmer

Time Limit: 1s

- 更简单的办法?
- $\left\lfloor \frac{x}{y} \right\rfloor = \left\lfloor \frac{y}{x} \right\rfloor = 1$
- 输入为0?
- 给每个输入都加1即可

Queries on Young Machine

Time Limit: 1s

- 定义个抽象计算机，处理32位带符号整数
- 存储：一个含 128 个位置的内存
- 程序：中缀表达式
- 程序输入：程序开始运行时内存0位置的值
- 程序输出：程序返回值
- 支持的运算：

Queries on Young Machine

Time Limit: 1s

调用格式	作用
x[hex_str]	返回十六进制数[hex_str]的值
<[pos]	然后返回内存的第pos号位置存储的值
>[pos][key]	内存的第pos个位置修改为key 返回key
?[expr0][expr1]	若内存第0个位置的值为0则执行[expr0]并返回，否则执行[expr1]并返回
|[expr0][expr1]	返回[expr0] | [expr1]
&[expr0][expr1]	返回[expr0]&[expr1]
^[expr0][expr1]	返回[expr0]^[expr1]
l[expr0][expr1]	返回[expr0]<<[expr1]
r[expr0][expr1]	返回[expr0]>>[expr1]
t[expr]	返回r的32位二进制表示中后缀0的个数
h[expr]	返回r的32位二进制表示中前缀0的个数

Queries on Young Machine

Time Limit: 1s

- 依次给出n个要求
- 每个要求形如(V, R)，表示对于输入V，你的程序的输出应为R
- 要求维护程序满足已给出的所有要求
- 维护的方式：在程序最后插入/删除字母
- 限制：程序长度限制、修改次数限制、运行步数限制

Queries on Young Machine

Time Limit: 1s

- **语句连接**
- 合理选择与/或/异或运算实现
- **判断两数是否相等**
- 判断异或值是否为零
- `|&x0>x0^[num1][num2]?[expr1][expr2]`
- 思路一：依次比较输入与每个要求是否相等
- 查询效率$O(n)$ 太低

Queries on Young Machine

Time Limit: 1s

- 还能做什么？
- 要在庞大的输入集合中快速定位一个数，二分查找？
- **比较大小**

1. 取[num1]和[num2]的异或值x
2. 取x的前导零个数h
3. 将[num1]左移h位再右移31位，记为r
4. 若r = 0则执行[expr1]，否则执行[expr2]

- `|&x0>x0rl[num1]h^[num1][num2]x1F?[expr1][expr2]`

Queries on Young Machine

Time Limit: 1s

- 至此我们可以解决静态版本的问题
- 如何维护新加入的要求?
- 思路二：插入一个数就全部重建
- 程序删改次数$O(n^2)$ 过多
- 思路三：分块 - 插入一个数之后局部重建一个小块，积累到$\sqrt{n}$个元素再重建整个程序
- 程序删改次数$O(n^{1.5})$ 仍过多

Queries on Young Machine

Time Limit: 1s

- 思路四：分块 - 插入一个数之后先使用思路一来维护当前块，积累到$\sqrt{n}$个元素再重建当前块
- 查询效率$O(\sqrt{n}\log n)$ 太低

- 反思问题
- 思路三：块合并得太勤
- 思路四：块合并得不够勤

Queries on Young Machine

Time Limit: 1s

- **更好的合并策略**
- 二进制分组
- 修改次数$O(n\log n)$
- 查询效率$O(\log^2 n)$

Queries on Young Machine

Time Limit: 1s

- 思考：如何编写高效的解释器

Parrots

Time Limit: 2s

- 写encoder和decoder两个程序
- encoder读入n个0到255之间的整数
- encoder输出不超过L个0到255之间的整数
- encoder的输出被打乱顺序后发送给decoder
- decoder需要还原出encoder所读入的n个数（次序也要一样）
- 构造编码方式
- $n \leq 64, L \leq 5n$

Parrots

Time Limit: 2s

- 考虑题目中所给的传递方式能够传递多少种信息
- 事实上，decoder收到的信息中，有用的信息是每个数出现的次数
- 最多能表示C_{L+255}^{255}种不同的信息
- 发送者可能发送256^n种不同的信息
- 要构造这256^n种发送信息到C_{L+255}^{255}种接收信息的单射
- 可以利用字典序排名
- Rank与发送/接收序列之间的转换通过简单DP完成

Saveit

Time Limit: 2s

- 给定一个n个点m条边的边的无向图，权均为1
- 给定一个值H
- 写encoder和decoder两个程序
- encoder可以获知这个图的信息和H的值
- 需将图的信息编码成一个二进制串
- decoder只能获知这串二进制串和H
- 要输出结点1到H与结点1到n两两之间的最短路

Saveit

Time Limit: 2s

- $n \leq 10^3, m \leq \frac{n \cdot (n-1)}{2}, H \leq 36$
- 二进制串长度不超过$7 \cdot 10^4$

Saveit

Time Limit: 2s

- 直接发整个图过去?
- 只知道所求的nH个最短路的信息是无法推知图的连边情况的
- 发送了过多信息
- 直接发H棵最短路树/H个最短路数组过去?
- 没有考虑这H棵树/H组距离之间的关联

Saveit

Time Limit: 2s

- 事实上，如果图中有边(u, v)，那么$\left|d_{i,u} - d_{i,v}\right| \leq 1$
- 我们发送了以1为根的最短路树后
- 发送2到其他点的最短路时
- 只需发送树上相邻两点的距离差即可
- 这个距离差只有-1,0,1三种取值
- 于是我们需要发送的信息是一棵树加上$H - 1$组距离差
- 可以在题目要求范围内实现

Last Supper

Time Limit: 2s

- 有n个数$0 \ldots n-1$
- 有k个槽作为缓存，最开始存放了$0 \ldots k-1$这些数
- 现在有n个请求，每次需要拿一个数
- 若请求的这个数在槽里，则称缓存命中
- 否则需要从不知道什么地方拿来这个数，并替换掉槽里的某一个数
- 已知请求列表，构造替换策略，使得命中次数最多

Last Supper

Time Limit: 2s

- 好吧我承认原题不是这样子的……
- 原题中已经给出了最优策略：
- 每次将槽中下一次被请求时间最晚的数替换掉
- 现在要求写两个程序
- 程序一知道请求序列，要生成一个摘要发给程序二
- 程序二需要拿着这个摘要，以最优的方式在线处理请求
- （最优是指，即使在预知请求序列的情况下命中次数也不可能更多）

Last Supper

Time Limit: 2s

- 要求尽量减小摘要长度
- 满分条件：摘要长度不超过$2n$ bit

Last Supper

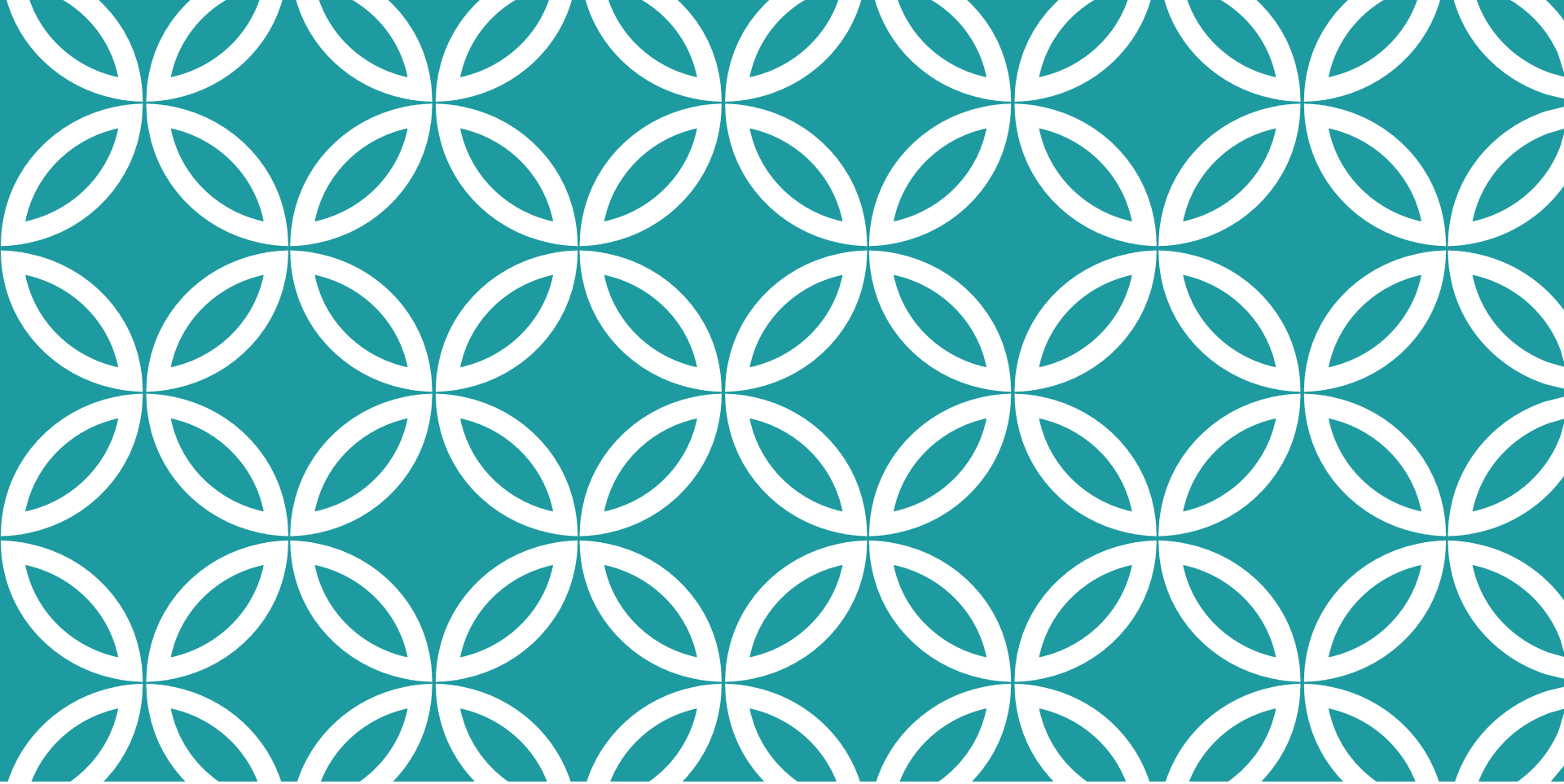
Time Limit: 2s

- 三个naive的思路
- 思路一：传输操作列表
- 思路二：传输每次被替换的位置
- 思路三：对一开始的每个数和之后加进来的每个数，传输它被替换掉的时间

Last Supper

Time Limit: 2s

- 有必要传输下次请求的时间么？
- 传输下次请求的时间是为了跟其他数的这个值进行比较，决定是否能丢弃它
- 那我们直接传输在下一次请求是否能丢弃它不是更好？
- 事实上，只要知道在下次请求之前能够丢弃它，什么时候丢弃是无关紧要的
- 于是我们只需要传输$n + k$个bit即可



构造策略

不限制可用操作

Professor Monotonic's Network

Time Limit: 3s

- 给定一个包含n个变量，m个比较器的比较网络
- 判定该网络是否是排序网络

- $1 \leq n \leq 12, 0 \leq m \leq 150$

Professor Monotonic's Network

Time Limit: 3s

- 排序网络的0-1原则
- 如果比较网络对所有的01序列能排序，则它是一个排序网络
- 证明?

Professor Monotonic's Network

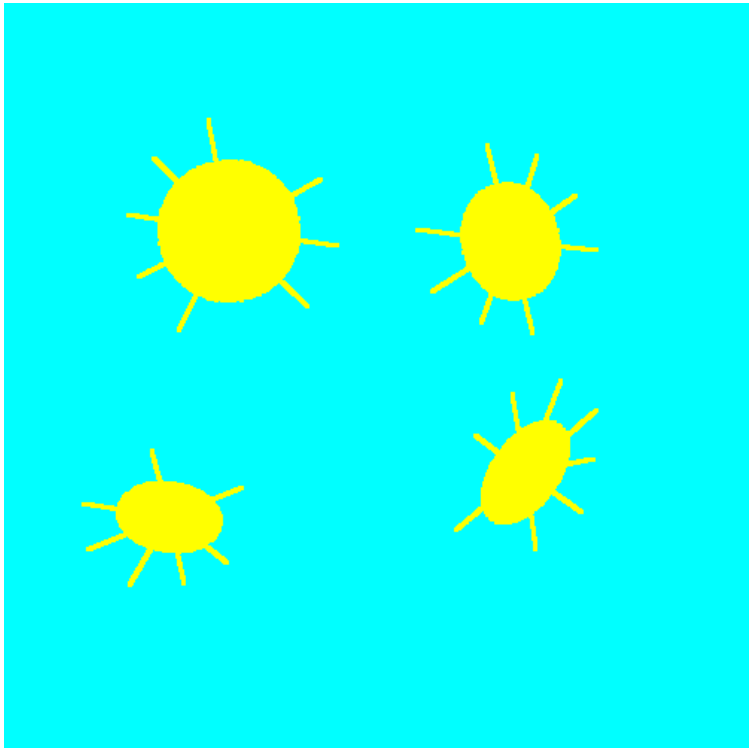
Time Limit: 3s

- 逆否命题：如果它不是排序网络，一定存在一个01序列它不能排序
- 任意找出一组不能被它排序的数组
- 设经过该比较网络运行结果中存在逆序对$(x, y)(x < y)$
- 把数组中$\geq y$的替换为1，其他替换为0
- 由于各数之间大小关系不变（不严格）
- 新数组与原数组进行的交换是相同的
- 得到了一个不能排序的01串

Suns and Rays

Time Limit: 3s

- 太阳：椭圆以及从这个椭圆发散出去的一些线段



Suns and Rays

Time Limit: 3s

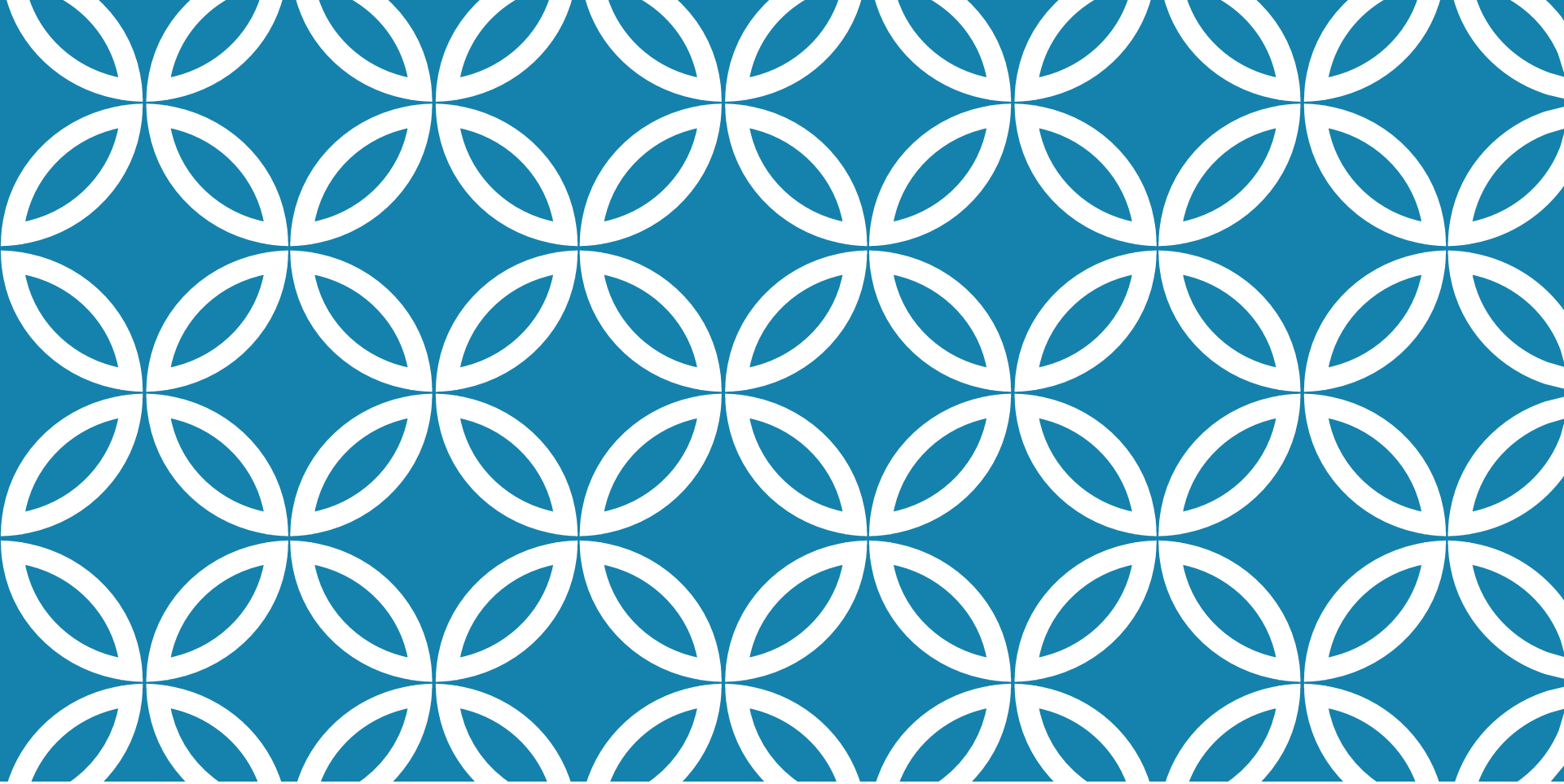
- 给定一张$h \times w$的位图
- 求出图中有多少个太阳
- 以及每个太阳上有多少条发散出去的线段

- $h, w \leq 1600$

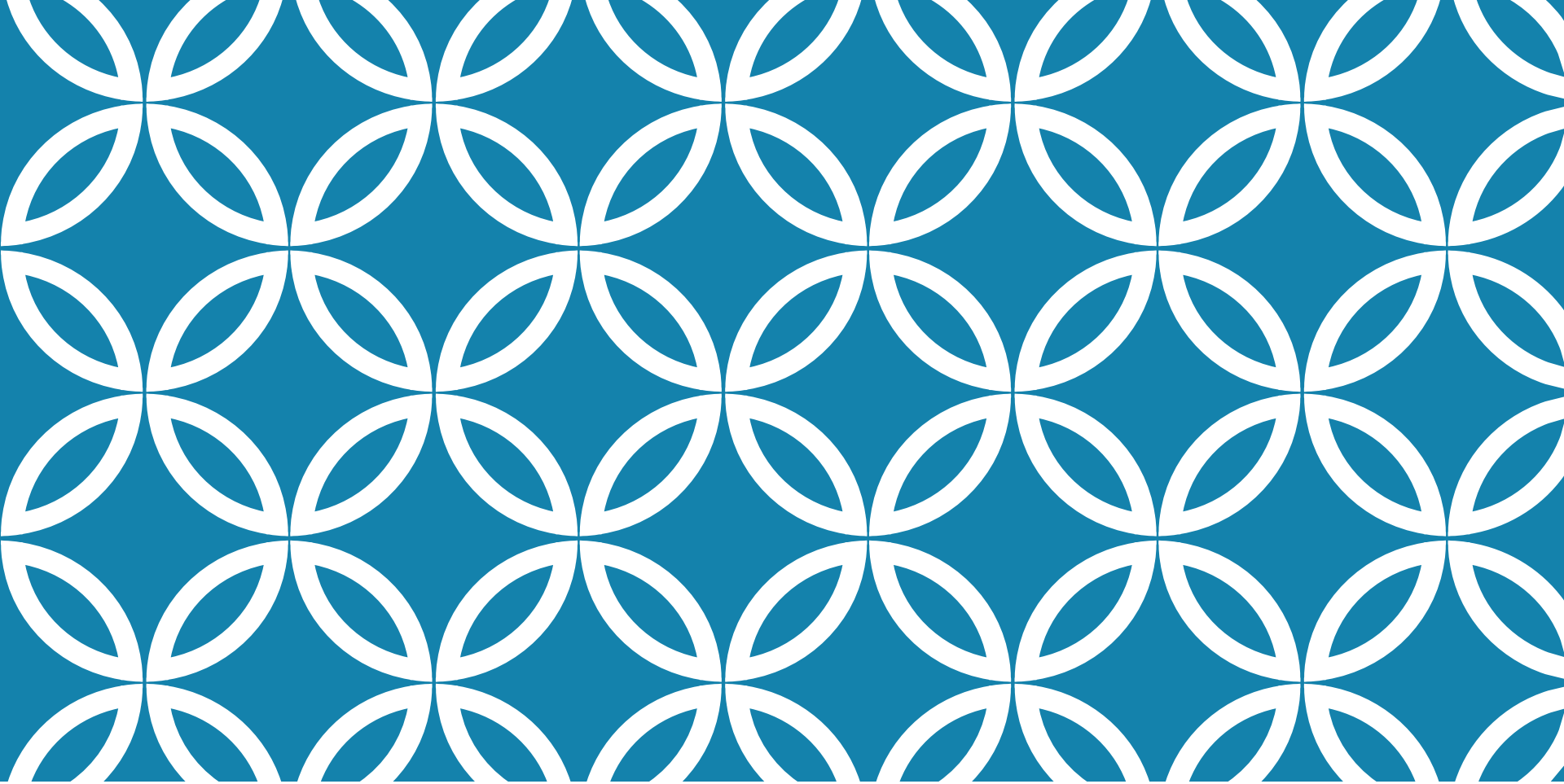
Suns and Rays

Time Limit: 3s

- 方法1
- 方法2
- ……



构造 | 终



感谢大家的支持 | 终