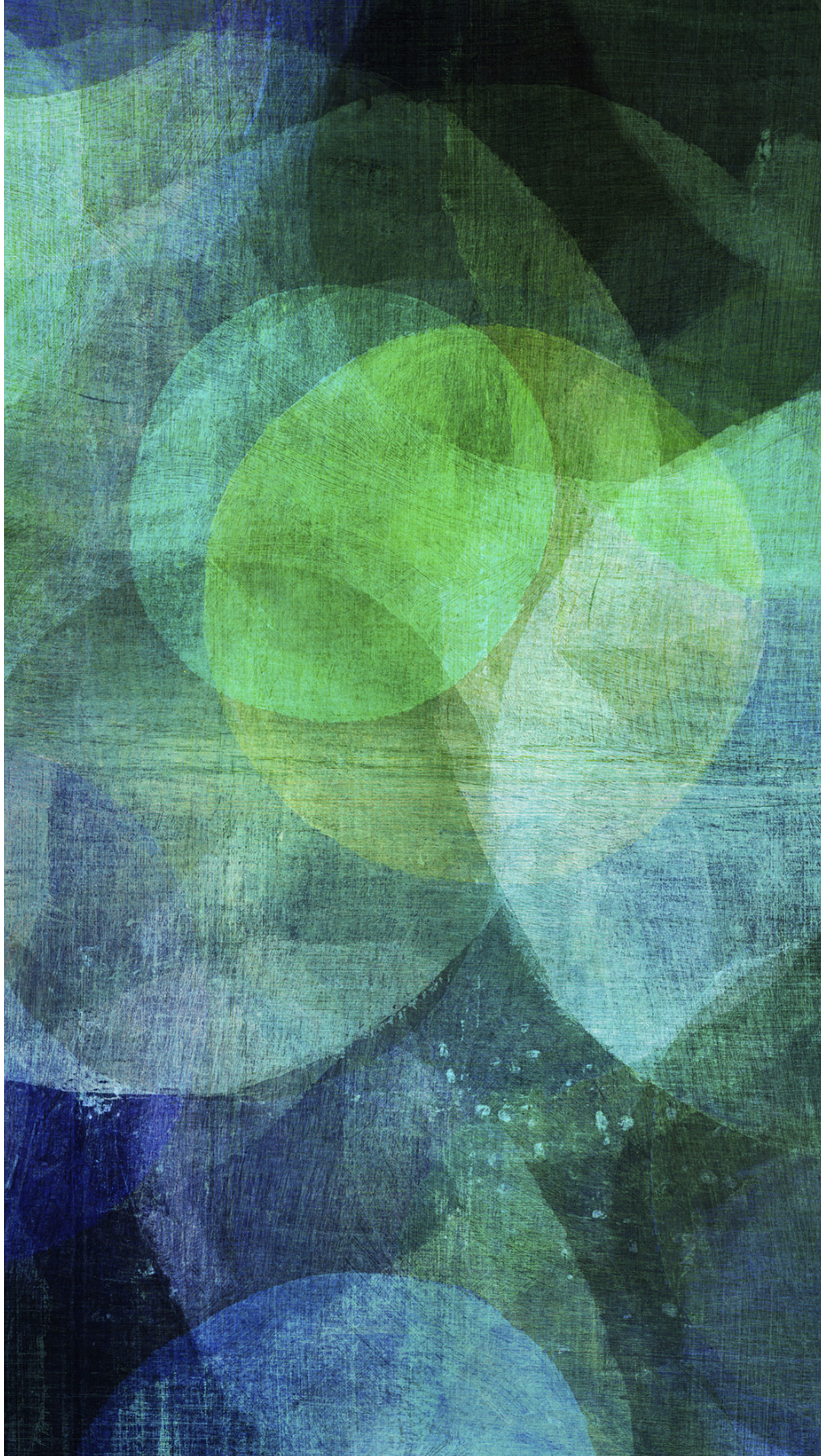
最大流最小割

最大流
DINIC算法

首先，学会一份效率较高的模版

- 最大流的求法用一句话解释就是带反悔的贪心，不停的去找增广路径，直到找不到为止
- 带反悔的贪心通过反向流去实现
- dinic：理论复杂度$n^2 * m$, 但是经过优化之后速度非常可观。

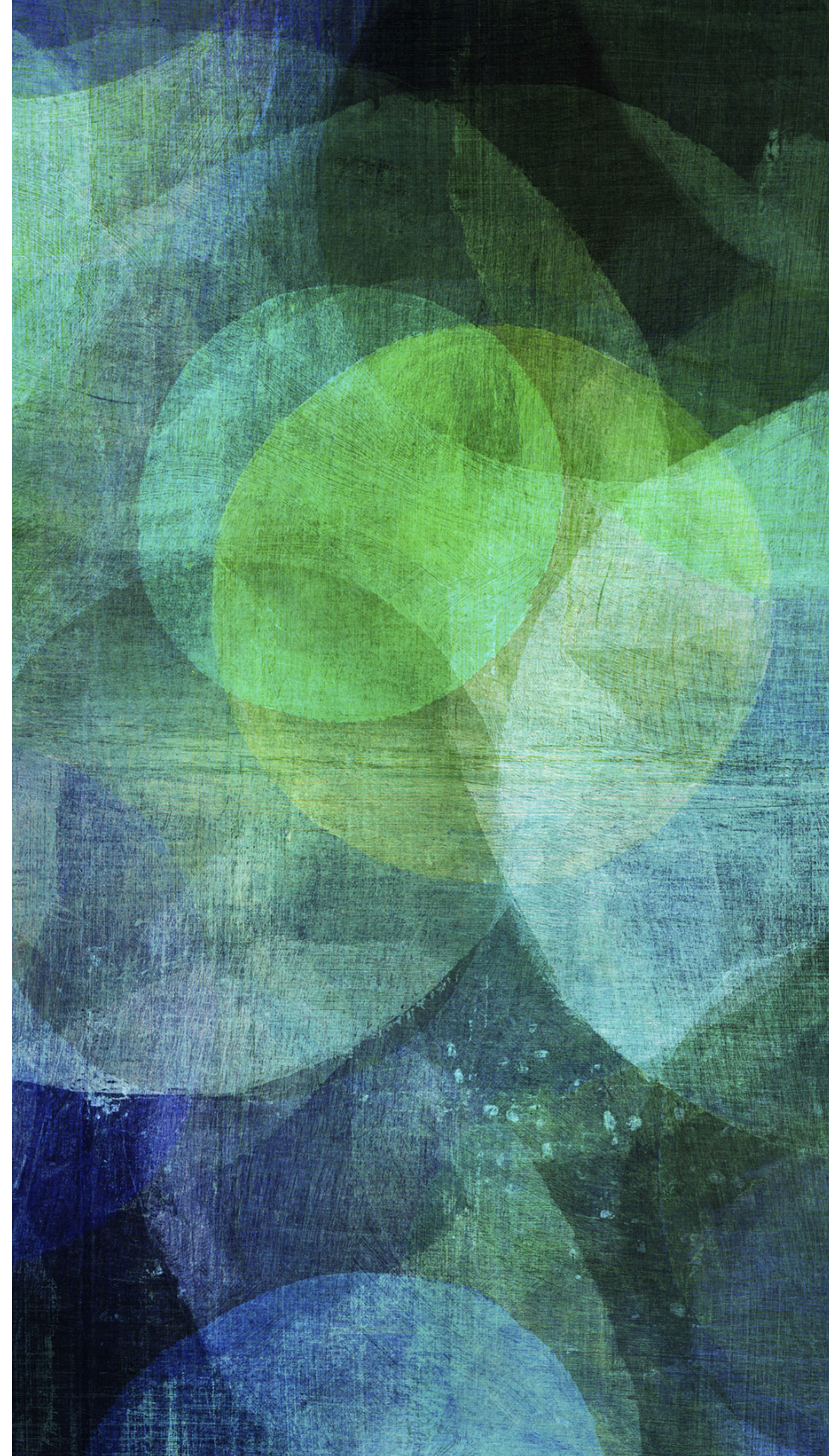
最小割



为什么最大流等于最小割

- 1： 可以观察到任意一个流小于等于任意一个割，因为割是容量之和，割对应的所有流都会小于割
- 2: 我们先跑一遍最大流，然后此时从s到t已经没有增广路了，我们把s能到的点集表示为S，到不了的点集表示为T，那么从S连向T的所有的边集必然是满流的。这些满流的边可以作为一个割，所以我们构造出了一个流等于割的例子
- 3: 结合1，2我们发现最大流＝最小割

应用



例题1：BZOJ1066 蜥蜴（拆点最大流）

- 在一个r行c列的网格地图中有一些高度不同的石柱，一些石柱上站着一些蜥蜴，你的任务是让尽量多的蜥蜴逃
- 到边界外。 每行每列中相邻石柱的距离为1，蜥蜴的跳跃距离是d，即蜥蜴可以跳到平面距离不超过d的任何一个石
- 柱上。石柱都不稳定，每次当蜥蜴跳跃时，所离开的石柱高度减1（如果仍然落在地图内部，则到达的石柱高度不
- 变），如果该石柱原来高度为1，则蜥蜴离开后消失。以后其他蜥蜴不能落脚。任何时刻不能有两只蜥蜴在同一个
- 石柱上。

解法:

- 将每个柱子拆成X Y两个点，两个点之间连的流量为柱子的高度，用来限制能跳到这个柱子的蜥蜴的数量，然后如果一个柱子i可以到达另一个柱子j，那么Yi就可以向Xj连边inf
- 再构造两个超级源点s与超级汇点t
- s向所有的蜥蜴的初始位置连边1
- 所有能跳出网格的柱子向t连边1
- 跑最大流，就是最多能出去的蜥蜴的数量

例题2: 最小割唯一性判定 ZOJ 2587

- 给你一个流量网络，判断最小割是否唯一

解法:

- 跑一遍最大流之后，在残留网络中S能搜到的点集以及能搜到T的点集恰好等于全集就代表唯一

例题3 BZOJ 1797 对于每条边都判断唯一性

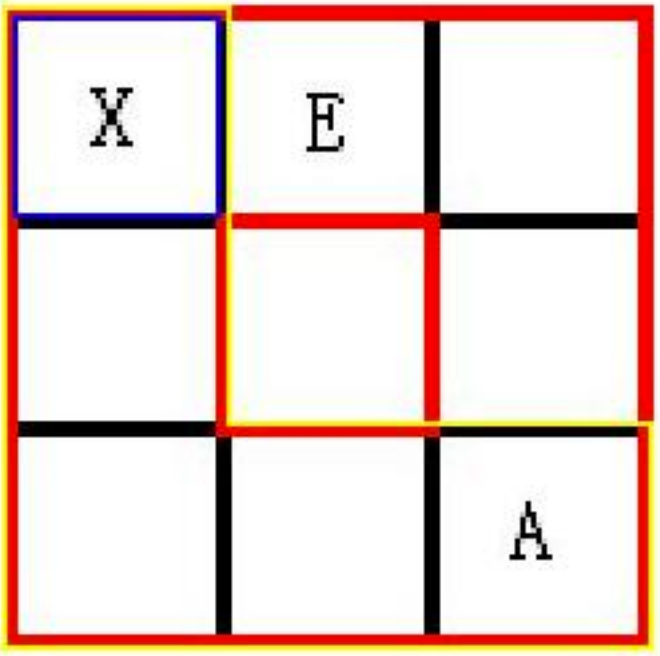
- 对于每条单向道路需要判断
- 问题一：是否存在一个最小代价路径切断方案，其中该道路被切断?
- 问题二：是否对任何一个最小代价路径切断方案，都有该道路被切断?
- 现在请你回答这两个问题。

解法：

- 在残余网络上跑tarjan求出所有SCC，记id[u]为点u所在SCC的编号。显然有id[s]!=id[t]（否则s到t有通路，能继续增广）。
- 对于任意一条满流边(u,v)，(u,v)能够出现在某个最小割集中，当且仅当id[u]!=id[v]；
- 对于任意一条满流边(u,v)，(u,v)必定出现在最小割集中，当且仅当id[u]==id[s]且id[v]==id[t]。
- ①将每个SCC缩成一个点，得到的新图就只含有满流边了。那么新图的任一s-t割都对应原图的某个最小割，从中任取一个把id[u]和id[v]割开的割即可证明。
- ②假设将(u,v)的边权增大，那么残余网络中会出现s->u->v->t的通路，从而能继续增广，于是最大流流量（也就是最小割容量）会增大。这即说明(u,v)是最小割集中必须出现的边。

例题4: ZOJ 3475（建图题）

- 在一个大陆上，有一个国家X，也有一些国家E，有国家X的同盟国A，X和A为了抵挡E的进攻，决定建造城墙来和国家E隔开，求建造这些城墙的最小值。A最多只有5个，每围一个A，可以获得一定利益
- 此题的读入方式需要仔细点


X
E
A

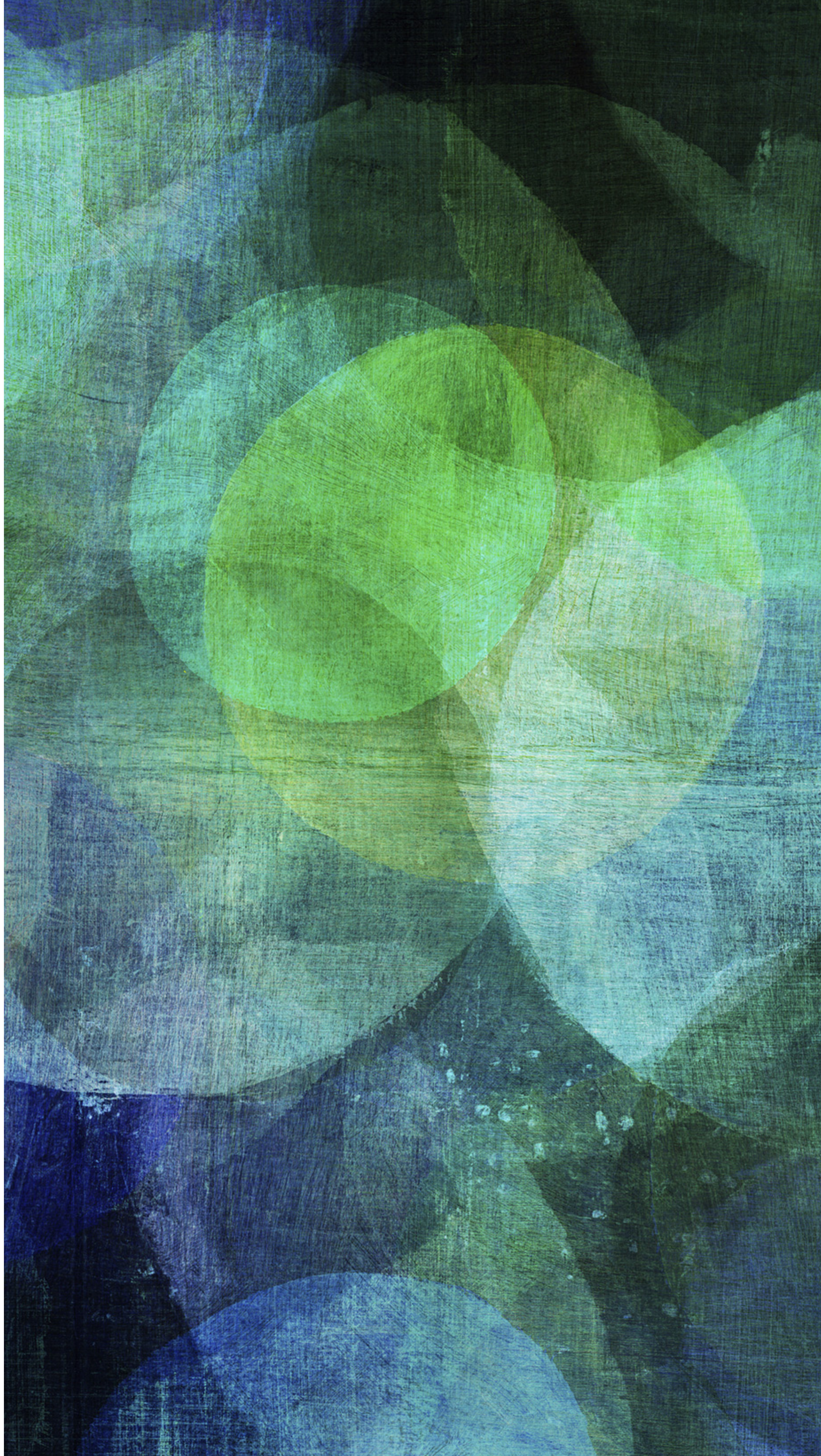
解法:

- 首先可以2^5暴力枚举选择哪些A一起围起来，建立超级源S，向X以及所有的A连inf的边，所有的E以及边界向超级汇点T连inf的边，相邻两个格子连边，流量为围墙的建造代价
- 我们发现此时求一遍最小割即可得到答案

例题5: 网络流与二分图

- 二分图最大匹配，可以改成从源点向所有的左部点连inf流量，所有右部点向汇点连inf流量，中间的边正常连， 最大流就是最大匹配

上下界网络流



分类

- 1：无源汇可行流 sgu 194
- 2：有源汇可行流 poj 2396
- 3：有源汇最大流 zoj 3496
- 4：有源汇最小流 hdu 3157

无源汇可行流

- 假设一条边的上界流为R 下界流为L，则 R-L为自由流，意思就是只要流了L，接下来要流多少随便。
- 无源汇的上下界可行流是指一种方案。流是循环在流的，每个点都满足流入等于流出。我们再细分一下就是sigma(进来的下界流) + sigma(进来的自由流) = sigma(出去的下界流)+sigma(出去的自由流)
- 假设我们要将这个网络流问题转换成普通的最大流问题，那么首先就是要把下界去掉，即下界为0。 去掉后要把下界的因素叠加在原图中，怎么叠加呢? 假设一个点进来的下界流总和减去出去的下界流总和为M，如果M为正，表示进来的自由流要比出去的自由流 少 M ，所以我们人为的向这个点补充进M（想想为什么）。如果M为负，也是类似的，从当前点额外地流出去M。 补充进流量和流出去需要建一个源点，一个汇点。
- 为什么这样子建好图求最大流后如果源点流出的边满流就有解?
- 想想看，如果源点流出的边都能满流，意味着什么? ? 我们可以把中间的网络看成一个黑箱，源点是入口，汇点是出口，不管中间是循环的流也好，或者是直接经过一条链到达汇点也好，我们不关心，我们关心的只是进去的流能否流出来，其实也就是黑箱里的网络能否支持流进来的流。如果能够支持，说明原网络能够支持这么些必须要流的下界限制。也就意味着原网络存在一个方案喽！

有源汇最大流

- 有源汇上下界的最大流的做法是：由汇点T向源点S建边（上界为INF，下界为0），（这一步非常犀利，等一下解释为啥么犀利），转换成无源汇的网络流问题，用上面的方法判断是否有解，如果有解的话，再跑一次源点S到汇点T的最大流，现在的最大流就是答案啦。。。
- 刚才那一步犀利犀利在只建了一条边就将原图转换成另一个已经可以解决的问题了
- 第一次跑完最大流的时候其实就可以求解网络中的一个可行流的流量了，这个流量其实就是最后加进去的那条边的反向流。
- 为什么最后再跑一遍最大流就能求得答案?
- 因为第一遍跑的是可行流，可能还有些边有余力可以再流一流，所以再跑一遍，原来那个可行流的流量记录在了T ->S的 反向流中，因次这个流再流一次就流回去了，所以再流一次就是答案！

有源汇最小流

- 先同无源汇，构造附加网络，求ss->tt的最大流，然后添加一条从t->s（0，inf），再从ss->tt跑最大流，t->s的流量就是答案
- 感性理解： 第一遍最大流释放能力，第二遍最大流 验证是否可行的同时让尽可能多的流量回来

例题： POJ 2396

- 有一个矩阵，告诉你每行的和以及每列的和，还有一些限制，综合起来其实就是告诉你某个数a[i][j]的范围，不过要自己去求。最后问你是否存在这样一个矩阵，有的话输出。

解法:

- 上下界可行流，但要先拆点，1~n*m表示矩阵中的点，n*m+1~到2*n*m表示拆成的点，2*n*m+1~2*n*m+m表示每一行，2*n*m+m+1~2*n*m+m+n表示每一列，那每行的和 每列的和可以通过新建源点，汇点来限制了，这些边的上下界都是一个定值，其余的就比较简单了，套个有源汇的上下界可行流就ok了。

- 最后再总结一下建图方法，方便现场赛可以快速的回忆起来。
- 1：无源汇的可行流： 新建源点，汇点，M[i]为每个点进来的下界流减去出去的下界流，如果M[i]为正，由源点向改点建M[i]的边，反之，由该点向汇点建M[i]的边，原图中的边为每条边的上界建去下界。跑一遍最大流，每条边的流量加上下界流就是答案。
- 2：有源汇的最大流： 从汇点向源点建一条容量为INF的边，用上面的方法判断是否有解，有解就再跑一遍从原图中源点到汇点的最大流
- 3：有源汇的最小流：先在附加网络跑一遍最大流，然后连上从汇点到源点的边，再按照1的方法做就好了