Graph theory in OI

July 6, 2018

Preface

- 课件按照知识-技巧-应用的顺序去组织。

Preface

- 课件按照知识-技巧-应用的顺序去组织。
- 主要侧重于这些知识在 OI 中的应用而不是这些知识本身。

Preface

- 课件按照知识-技巧-应用的顺序去组织。
- 主要侧重于这些知识在 OI 中的应用而不是这些知识本身。
- 所有例题来自于我高中时期做过的题目，水平有限，如果各位见过这些题目欢迎发言。

Preface

- 课件按照知识-技巧-应用的顺序去组织。
- 主要侧重于这些知识在 OI 中的应用而不是这些知识本身。
- 所有例题来自于我高中时期做过的题目，水平有限，如果各位见过这些题目欢迎发言。
- 会适当补充在 OI 界尚不普及的技巧来开阔眼界，少数技巧不一定实用，请在比赛中小心使用。

cf 715B

- $n \leq 1000, m \leq 10^4$ 的无向图，一些边的边权你可以任意决定为一些正数，使得 S 到 T 的最短路长度恰好为 L。

cf 715B

- 注意我们只要保存每个边权都为 1 的时候的最短路中的边就可以了

cf 715B

- 注意我们只要保存每个边权都为 1 的时候的最短路中的边就可以了
- 边数缩小到 $O(n)$ 条，那么直接修改每个边的边权使得这条路的距离为 L, 然后 check 一下。如果最短路还是小于 L, 那么可以把这个边删去。$O(nmlogn)$

cf 715B

- 注意我们只要保存每个边权都为 1 的时候的最短路中的边就可以了
- 边数缩小到 $O(n)$ 条，那么直接修改每个边的边权使得这条路的距离为 L, 然后 check 一下。如果最短路还是小于 L, 那么可以把这个边删去。$O(nmlogn)$
- 更好的做法?

cf 715B

- 注意我们只要保存每个边权都为 1 的时候的最短路中的边就可以了
- 边数缩小到 $O(n)$ 条，那么直接修改每个边的边权使得这条路的距离为 L, 然后 check 一下。如果最短路还是小于 L，那么可以把这个边删去。$O(nmlogn)$
- 更好的做法?
- 我们可以把一个前缀赋为 $x+1$，一个后缀赋为 x，连续性单调性证明可行。$O(nlogL)$

Multiplicative Approximate all pair shortest path

- 近似 APSP 等价于近似 $(min, +)-$product（也叫 distance product）.

Multiplicative Approximate all pair shortest path

- 近似 APSP 等价于近似 $(min, +)-$product（也叫 distance product）.
- 如果我们能近似 distance product，那么就可以近似 APSP

Multiplicative Approximate all pair shortest path

- 近似 APSP 等价于近似 $(min, +)-$product（也叫 distance product）.
- 如果我们能近似 distance product，那么就可以近似 APSP
- 近似常用的办法：把权值向上取整为 $O(\log(n))$ 种。

Multiplicative Approximate all pair shortest path

- 近似 APSP 等价于近似 $(min, +)-$product（也叫 distance product）.
- 如果我们能近似 distance product，那么就可以近似 APSP
- 近似常用的办法：把权值向上取整为 $O(\log(n))$ 种。
- 变成多项式的矩阵乘法，用快速矩阵乘优化

Multiplicative Approximate all pair shortest path

- 近似 APSP 等价于近似 $(min, +)-$product（也叫 distance product）.
- 如果我们能近似 distance product，那么就可以近似 APSP
- 近似常用的办法：把权值向上取整为 $O(\log(n))$ 种。
- 变成多项式的矩阵乘法，用快速矩阵乘优化
- so easy

All pair shortest path

- 边权只为 $-1/0/1$ 的 APSP 我们能不能做的更快?

Approximate all pair shortest path

- 还能再优化吗？

Approximate all pair shortest path

- 还能再优化吗？
- 回到近似 APSP 的问题，我们做 $O(\log(n))$ 次 distance product，是默认了路径长度每次乘 2。

Approximate all pair shortest path

- 还能再优化吗?
- 回到近似 APSP 的问题，我们做 $O(\log(n))$ 次 distance product，是默认了路径长度每次乘 2。
- 每次乘 1.5 会怎么样?

Approximate all pair shortest path

- 还能再优化吗？
- 回到近似 APSP 的问题，我们做 $O(\log(n))$ 次 distance product，是默认了路径长度每次乘 2。
- 每次乘 1.5 会怎么样？
- 因为 $2x - 1.5x = 0.5x$ 会有一个 $O(x)$ 大小的交，同时又有至多 $O(n^2)$ 对路径

Approximate all pair shortest path

- 还能再优化吗？
- 回到近似 APSP 的问题，我们做 $O(\log(n))$ 次 distance product，是默认了路径长度每次乘 2。
- 每次乘 1.5 会怎么样？
- 因为 $2x - 1.5x = 0.5x$ 会有一个 $O(x)$ 大小的交，同时又有至多 $O(n^2)$ 对路径
- 随机 $O(\frac{n}{x}\log(n))$ 个点做 bridging set！矩阵的大小被缩成了长方形。

Approximate all pair shortest path

- 还能再优化吗？
- 回到近似 APSP 的问题，我们做 $O(\log(n))$ 次 distance product，是默认了路径长度每次乘 2。
- 每次乘 1.5 会怎么样？
- 因为 $2x - 1.5x = 0.5x$ 会有一个 $O(x)$ 大小的交，同时又有至多 $O(n^2)$ 对路径
- 随机 $O(\frac{n}{x}\log(n))$ 个点做 bridging set！矩阵的大小被缩成了长方形。
- 在 x 小的时候矩阵乘法，在 x 大的时候暴力。

cf 329E

笛卡尔坐标系上 n 个整点，求最大曼哈顿距离哈密顿回路。

$n \leq 10^5$

cf 329E

- 先考虑一维的情况，每条边的贡献为 $2i$

cf 329E

- 先考虑一维的情况，每条边的贡献为 $2i$
- 贡献放到点上就是每个点的贡献为 $2x$ 或者 $-2x$，需要是『单个』的合法括号序列

cf 329E

- 先考虑一维的情况，每条边的贡献为 $2i$
- 贡献放到点上就是每个点的贡献为 $2x$ 或者 $-2x$，需要是『单个』的合法括号序列
- 那么显然中位数左边为 $-2x$，中位数右边为 $2x$ 最优

cf 329E

- 先考虑一维的情况，每条边的贡献为 $2i$
- 贡献放到点上就是每个点的贡献为 $2x$ 或者 $-2x$，需要是『单个』的合法括号序列
- 那么显然中位数左边为 $-2x$，中位数右边为 $2x$ 最优
- 二维取两维中位数

cf 329E

- 先考虑一维的情况，每条边的贡献为 $2i$
- 贡献放到点上就是每个点的贡献为 $2x$ 或者 $-2x$，需要是『单个』的合法括号序列
- 那么显然中位数左边为 $-2x$，中位数右边为 $2x$ 最优
- 二维取两维中位数
- 左下等于右上，左上等于右下，但是不连通

cf 329E

- 改两个边把两个环并起来，交叉互换

cf 329E

- 改两个边把两个环并起来，交叉互换
- 考虑一个蝴蝶形。

cf 329E

- 改两个边把两个环并起来，交叉互换
- 考虑一个蝴蝶形。
- 所以取 $min(x_{n/2+1} - x_{n/2}, y_{n/2+1} - y_{n/2})$

cf 329E

- 还没做完，别忘了奇数的情况

cf 329E

- 还没做完，别忘了奇数的情况
- 如果两个线交在了两个点上，那么我们注意这两个点可以把两个环连通起来了

cf 329E

- 还没做完，别忘了奇数的情况
- 如果两个线交在了两个点上，那么我们注意这两个点可以把两个环连通起来了
- 如果两个线正好交在一个中心的一个点上，那么这个点可以用来做那个交叉互换

趣题 1

- 有一个 n 个点的折线构成的山脉，两个人鹊巢相会，他们深爱着彼此，所以始终会保持着同一高度。

趣题 1

- 有一个 n 个点的折线构成的山脉，两个人鹊巢相会，他们深爱着彼此，所以始终会保持着同一高度。
- 请你输出他们能不能相会。

趣题 1

- 有一个 n 个点的折线构成的山脉，两个人鹊巢相会，他们深爱着彼此，所以始终会保持着同一高度。
- 请你输出他们能不能相会。
- $n \leq 10^7$

趣题 1

- 先把高度全部离散化

趣题 1

- 先把高度全部离散化
- 每个点对表示一个两个人的状态 (x, y)

趣题 1

- 先把高度全部离散化
- 每个点对表示一个两个人的状态 (x, y)
- 每个点对度都为偶数

趣题 1

- 先把高度全部离散化
- 每个点对表示一个两个人的状态 (x, y)
- 每个点对度都为偶数
- 偏偏起始状态度为奇数?

趣题 1

- 先把高度全部离散化
- 每个点对表示一个两个人的状态 (x, y)
- 每个点对度都为偶数
- 偏偏起始状态度为奇数?
- 终止状态度数也为奇数! 一定可以走到!

Shortest path mod k

- $n \leq 10^5$ 的无向连通图求两点间模 $k \leq 10^9$ 意义下的最短路，可以重复经过边和点

Shortest path mod k

- $n \leq 10^5$ 的无向连通图求两点间模 $k \leq 10^9$ 意义下的最短路，可以重复经过边和点
- author : jcvb

Shortest path mod k

- $n \leq 10^5$ 的无向连通图求两点间模 $k \leq 10^9$ 意义下的最短路，可以重复经过边和点
- author：jcvb
- 『我尝试用我的人工智能程序来运行最短路算法，可它似乎出了一点 BUG。它一直在一条正权边上来回穿梭，试图通过超过有符号 32 位长整形所能表示的最大正整数的方式来达到总长度最小化。——p 妈』

Shortest path mod k

- 一样的道理，我们可以反复走同一条边

Shortest path mod k

- 一样的道理，我们可以反复走同一条边
- mod 奇数 k 可以反复走 k 次就消掉了

Shortest path mod k

- 一样的道理，我们可以反复走同一条边
- mod 奇数 k 可以反复走 k 次就消掉了
- mod 偶数，CRT 考虑每个 p^q，生成出 $p^{q'}$ 的所有倍数

Shortest path mod k

- 一样的道理，我们可以反复走同一条边
- mod 奇数 k 可以反复走 k 次就消掉了
- mod 偶数，CRT 考虑每个 p^q，生成出 $p^{q'}$ 的所有倍数
- 2^q 是个例外，判定二分图

Shortest path mod k

- 一样的道理，我们可以反复走同一条边
- mod 奇数 k 可以反复走 k 次就消掉了
- mod 偶数，CRT 考虑每个 p^q，生成出 $p^{q'}$ 的所有倍数
- 2^q 是个例外，判定二分图
- 最后其实就等于是 0 或者 gcd

趣题 2

- 一个三角形的三角剖分的三个角被染成了三种颜色。

趣题 2

- 一个三角形的三角剖分的三个角被染成了三种颜色。
- 三角形边上的点的颜色一定要与两端点中的一个相同

趣题 2

- 一个三角形的三角剖分的三个角被染成了三种颜色。
- 三角形边上的点的颜色一定要与两端点中的一个相同
- 试证明内部一定存在一个同色三角形

趣题 2

- 先规定每次对着红-蓝边走，保证蓝色在你的右手边

趣题 2

- 先规定每次对着红-蓝边走，保证蓝色在你的右手边
- 走不动就找到了同色三角形

趣题 2

- 先规定每次对着红-蓝边走，保证蓝色在你的右手边
- 走不动就找到了同色三角形
- 奇偶性分析一定存在一个路径不会转出来

bzoj4061

- 一个 $n \leq 10^5$ 个点 $m \leq 3 * 10^5$ 个边的无向图

bzoj4061

- 一个 $n \leq 10^5$ 个点 $m \leq 3 * 10^5$ 个边的无向图
- 新建一个点，确定它到每个边的路径长度，非负可以不是整数

bzoj4061

- 一个 $n \leq 10^5$ 个点 $m \leq 3 * 10^5$ 个边的无向图
- 新建一个点，确定它到每个边的路径长度，非负可以不是整数
- 使得每个点到 1 和 2 的最短路都不经过他

bzoj4061

- 一个 $n \leq 10^5$ 个点 $m \leq 3 * 10^5$ 个边的无向图
- 新建一个点，确定它到每个边的路径长度，非负可以不是整数
- 使得每个点到 1 和 2 的最短路都不经过他
- 最小化它到每个点的距离的和

bzoj4061

- 先求出每个点到 1 和 2 的最短路

bzoj4061

- 先求出每个点到 1 和 2 的最短路
- x_i 表示这个点到 i 的距离，以 1 为例可以列出方程 $x_1 + x_i \geq k$ 和 $x_i + k \geq x_1$

bzoj4061

- 先求出每个点到 1 和 2 的最短路
- x_i 表示这个点到 i 的距离，以 1 为例可以列出方程 $x_1 + x_i \geq k$ 和 $x_i + k \geq x_1$
- 显然在满足方程的情况下每个点的距离 x_i 越小越好，从而自然就满足了 $x_i \leq x_1 + k$。

bzoj4061

- 先求出每个点到 1 和 2 的最短路
- x_i 表示这个点到 i 的距离，以 1 为例可以列出方程 $x_1 + x_i \geq k$ 和 $x_i + k \geq x_1$
- 显然在满足方程的情况下每个点的距离 x_i 越小越好，从而自然就满足了 $x_i \leq x_1 + k$。
- $x_i = max(|x_1 - k_1|, |x_2 - k_2|)$

bzoj4061

- 先求出每个点到 1 和 2 的最短路
- x_i 表示这个点到 i 的距离，以 1 为例可以列出方程 $x_1 + x_i \geq k$ 和 $x_i + k \geq x_1$
- 显然在满足方程的情况下每个点的距离 x_i 越小越好，从而自然就满足了 $x_i \leq x_1 + k$。
- $x_i = max(|x_1 - k_1|, |x_2 - k_2|)$
- 最小化 $\sum x_i$，这是个切比雪夫距离，转 45 度变成曼哈顿距离

bzoj4061

- 先求出每个点到 1 和 2 的最短路
- x_i 表示这个点到 i 的距离，以 1 为例可以列出方程 $x_1 + x_i \geq k$ 和 $x_i + k \geq x_1$
- 显然在满足方程的情况下每个点的距离 x_i 越小越好，从而自然就满足了 $x_i \leq x_1 + k$。
- $x_i = max(|x_1 - k_1|, |x_2 - k_2|)$
- 最小化 $\sum x_i$，这是个切比雪夫距离，转 45 度变成曼哈顿距离
- 两维取中位数即可

Threshold method

- n 个点 m 条边的带权有向图，求从 s 到 t 的最大边权最小的路径。

Threshold method

- n 个点 m 条边的带权有向图，求从 s 到 t 的最大边权最小的路径。
- $O((n+m)\log(n))$

Threshold method

- 如果 $m = O(n)$ 呢?

Threshold method

- 如果 $m = O(n)$ 呢?
- 考虑拓展整体二分，每次设 k 个 threshold，$m_1, \ldots, m_k$

Threshold method

- 如果 $m = O(n)$ 呢?
- 考虑拓展整体二分，每次设 k 个 threshold，$m_1, \ldots, m_k$
- 我们把边权替换成对应的 1 到 k，$O(n + m + k)$ 地求出从 s 出发的最小瓶颈路。

Threshold method

- 如果 $m = O(n)$ 呢?
- 考虑拓展整体二分，每次设 k 个 threshold，$m_1, \ldots, m_k$
- 我们把边权替换成对应的 1 到 k，$O(n + m + k)$ 地求出从 s 出发的最小瓶颈路。
- 每次递归进 t 对应的区间。

Threshold method

- 如果 $m = O(n)$ 呢?
- 考虑拓展整体二分，每次设 k 个 threshold，$m_1, \ldots, m_k$
- 我们把边权替换成对应的 1 到 k，$O(n + m + k)$ 地求出从 s 出发的最小瓶颈路。
- 每次递归进 t 对应的区间。
- 具体来说，注意每次如果一个边两端的点不在一块，这个边一定会被抛弃。

Threshold method

- “把边权替换成对应的 1 到 k”二分又会带上 $\log(n)$

Threshold method

- “把边权替换成对应的 1 到 k”二分又会带上 $\log(n)$
- “注意每次如果一个边两端的点不在一块，这个边一定会被抛弃。”

Threshold method

- “把边权替换成对应的 1 到 k”二分又会带上 $\log(n)$
- “注意每次如果一个边两端的点不在一块，这个边一定会被抛弃。”
- 那么我们可以二分这个边在哪一块，这总复杂度是 $O(m\log(k))$

Threshold method

- “把边权替换成对应的 1 到 k”二分又会带上 $\log(n)$
- “注意每次如果一个边两端的点不在一块，这个边一定会被抛弃。”
- 那么我们可以二分这个边在哪一块，这总复杂度是 $O(m\log(k))$
- 但是递归下来之前，很多点已经有初始的答案了，这些答案不能每层 $O(n\log(k))$ 暴力。

Threshold method

- “把边权替换成对应的 1 到 k”二分又会带上 $\log(n)$
- “注意每次如果一个边两端的点不在一块，这个边一定会被抛弃。”
- 那么我们可以二分这个边在哪一块，这总复杂度是 $O(m\log(k))$
- 但是递归下来之前，很多点已经有初始的答案了，这些答案不能每层 $O(n\log(k))$ 暴力。
- 树分块！我们每 $\log(k)$ 一块。每个联通块内我们二分最小点的答案，然后把他和权值相同的点一波带走即可

Threshold method

- “把边权替换成对应的 1 到 k”二分又会带上 $\log(n)$
- “注意每次如果一个边两端的点不在一块，这个边一定会被抛弃。”
- 那么我们可以二分这个边在哪一块，这总复杂度是 $O(m\log(k))$
- 但是递归下来之前，很多点已经有初始的答案了，这些答案不能每层 $O(n\log(k))$ 暴力。
- 树分块！我们每 $\log(k)$ 一块。每个联通块内我们二分最小点的答案，然后把他和权值相同的点一波带走即可
- 之后的点如果比他所在的块大，一定会有一条边横跨在两块之间！

Threshold method

- $O(m \log(k) + m \log(k)/\log(n)) = O(m\sqrt{\log(n)})$

无向图中的最小环

- 给一个 $n \leq 100$ 个点的无向图，请你求出过无向图中每个点的最小环？

无向图中的最小环

- 给一个 $n \leq 100$ 个点的无向图，请你求出过无向图中每个点的最小环？
- 分治 +floyd

无向图中的最小环

- 给一个 $n \leq 100$ 个点的无向图，请你求出过无向图中每个点的最小环？
- 分治 +floyd
- 二进制分组 + dijkstra

Nightmare of (some) OIers : Dynamic graph connectivity

- 给定一个 n 个节点的图，支持加边、删边，求两点间是否联通。

Nightmare of (some) OIers : Dynamic graph connectivity

- 允许离线？对时间轴建线段树、LCT 维护只有加边的生成树

Nightmare of (some) OIers : Dynamic graph connectivity

- 允许离线？对时间轴建线段树、LCT 维护只有加边的生成树
- 强制在线？

Nightmare of (some) OIers : Dynamic graph connectivity

- 允许离线？对时间轴建线段树、LCT 维护只有加边的生成树
- 强制在线？
- 先来看一道 uoj 的题目

共价大爷游长沙

- 长沙市的交通线路可以抽象成为一个 n 个点 $n-1$ 条边的无向图，点编号为 1 到 n，任意两点间均存在恰好一条路径，显然两个点之间最多也只会有一条边相连。

共价大爷游长沙

- 长沙市的交通线路可以抽象成为一个 n 个点 $n-1$ 条边的无向图，点编号为 1 到 n，任意两点间均存在恰好一条路径，显然两个点之间最多也只会有一条边相连。
- 有一个包含一些点的可重集合 S，共价大爷的旅行路线是这样确定的：每次他会选择某一对点 (x,y) 并从 x 出发沿着唯一路径到达 y。

共价大爷游长沙

- 长沙市的交通线路可以抽象成为一个 n 个点 $n-1$ 条边的无向图，点编号为 1 到 n，任意两点间均存在恰好一条路径，显然两个点之间最多也只会有一条边相连。
- 有一个包含一些点的可重集合 S，共价大爷的旅行路线是这样确定的：每次他会选择某一对点 (x,y) 并从 x 出发沿着唯一路径到达 y。
- 现在小 L 想知道，如果他守在某一条边，是否一定能见到共价大爷。

共价大爷游长沙

- 长沙市的交通线路可以抽象成为一个 n 个点 $n-1$ 条边的无向图，点编号为 1 到 n，任意两点间均存在恰好一条路径，显然两个点之间最多也只会有一条边相连。
- 有一个包含一些点的可重集合 S，共价大爷的旅行路线是这样确定的：每次他会选择某一对点 (x,y) 并从 x 出发沿着唯一路径到达 y。
- 现在小 L 想知道，如果他守在某一条边，是否一定能见到共价大爷。
- 可能某个时刻某条边会断开，同时这个时刻一定也有某条新边会出现，S 也会不断加入新点对或者删除原有的点对。当然，小 L 也有可能在任何时候向你提出守在某一条边是否一定能见到共价大爷的问题。你能回答小 L 的所有问题吗？

共价大爷游长沙

- 每次等于加和删点对是路径 xor 上一个值

共价大爷游长沙

- 每次等于加和删点对是路径 xor 上一个值
- 改一条边的位置，等于是把对应联通块端点之间的路径 xor 上那个随机值

共价大爷游长沙

- 每次等于加和删点对是路径 xor 上一个值
- 改一条边的位置，等于是把对应联通块端点之间的路径 xor 上那个随机值
- 能否一定见到其实就是比较一下 hash 值即可

Nightmare of (some) Olers : Dynamic graph connectivity

- 我们来应用一样的想法!

Nightmare of (some) Olers : Dynamic graph connectivity

- 我们来应用一样的想法!
- 如果删除一条边的时候只有一条替代边?

Nightmare of (some) Olers : Dynamic graph connectivity

- 我们来应用一样的想法!
- 如果删除一条边的时候只有一条替代边?
- 联通块的端点 xor 和

Nightmare of (some) OIers : Dynamic graph connectivity

- 我们来应用一样的想法!
- 如果删除一条边的时候只有一条替代边?
- 联通块的端点 xor 和
- 如果有多条替代边如何还原出一条?

Nightmare of (some) Olers : Dynamic graph connectivity

- 我们来应用一样的想法!
- 如果删除一条边的时候只有一条替代边?
- 联通块的端点 xor 和
- 如果有多条替代边如何还原出一条?
- 维护 $O(\log(n))$ 个这样的结构，第 i 层的若干个以 $\frac{1}{2^c}$ 的概率在两端点 xor 上自己的编号

cf657 E

- $n, m \leq 3 * 10^5$ 的无向图, $q \leq 3 * 10^5$ 个询问

cf657 E

- $n, m \leq 3 * 10^5$ 的无向图, $q \leq 3 * 10^5$ 个询问
- 每次给你一个点的集合 V' 和额外的边集 E'

cf657 E

- $n, m \leq 3 * 10^5$ 的无向图, $q \leq 3 * 10^5$ 个询问
- 每次给你一个点的集合 V' 和额外的边集 E'
- 问是否 $\forall x, y \in V'$，在 $E \cup E'$ 中存在一条不经过重复边的路径先从 x 到 y 再从 y 到 x, 强制在线

cf657 E

- $n, m \leq 3 * 10^5$ 的无向图，$q \leq 3 * 10^5$ 个询问
- 每次给你一个点的集合 V' 和额外的边集 E'
- 问是否 $\forall x, y \in V'$，在 $E \cup E'$ 中存在一条不经过重复边的路径先从 x 到 y 再从 y 到 x，强制在线
- 缩双连通分量，之后求个虚树判断是否双连通即可

Korn

一个有 n 个点的无向图
求所有这样的点:
从它开始走，只要当前点有一条连出去的未访问过的边就要继续走，每次必须走一条之前没有访问过的边
如果从它开始走无论怎么走都会走出一条欧拉回路
就称它不可避
求所有不可避的点
$n \leq 10^5$

Korn

- 猜猜什么时候不可避?

Korn

- 猜猜什么时候不可避?
- 当且仅当所有环都经过这个点

Korn

- 猜猜什么时候不可避?
- 当且仅当所有环都经过这个点
- 充分性显然，必要性考虑把不经过这个点的环删去之后求欧拉回路

Korn

- 猜猜什么时候不可避?
- 当且仅当所有环都经过这个点
- 充分性显然，必要性考虑把不经过这个点的环删去之后求欧拉回路
- 怎么判定呢?

Korn

- 考虑删除这个点，所有环都过这个点的话留下的一定是森林

Korn

- 考虑删除这个点，所有环都过这个点的话留下的一定是森林
- 考虑删去这个点之后可能分成若干联通块

Korn

- 考虑删除这个点，所有环都过这个点的话留下的一定是森林
- 考虑删去这个点之后可能分成若干联通块
- 每个联通块是树的充要条件是恰有 n-1 条边

Korn

- 考虑删除这个点，所有环都过这个点的话留下的一定是森林
- 考虑删去这个点之后可能分成若干联通块
- 每个联通块是树的充要条件是恰有 n-1 条边
- 如果这个点不是割点，直接减去这个点的度数

Korn

- 考虑删除这个点，所有环都过这个点的话留下的一定是森林
- 考虑删去这个点之后可能分成若干联通块
- 每个联通块是树的充要条件是恰有 n-1 条边
- 如果这个点不是割点，直接减去这个点的度数
- 如果这个点是割点，枚举这个点连出的每条边并判断？

Korn

- 考虑删除这个点，所有环都过这个点的话留下的一定是森林
- 考虑删去这个点之后可能分成若干联通块
- 每个联通块是树的充要条件是恰有 n-1 条边
- 如果这个点不是割点，直接减去这个点的度数
- 如果这个点是割点，枚举这个点连出的每条边并判断？
- 怎么知道连到的每个联通块所包含的边数呢

Korn

- 考虑删除这个点，所有环都过这个点的话留下的一定是森林
- 考虑删去这个点之后可能分成若干联通块
- 每个联通块是树的充要条件是恰有 n-1 条边
- 如果这个点不是割点，直接减去这个点的度数
- 如果这个点是割点，枚举这个点连出的每条边并判断？
- 怎么知道连到的每个联通块所包含的边数呢
- 缩点双形成点双树，点双树中记录下子树边数和即可

hihocoder Challenge 28

- $n, k \leq 50, e_w \leq 10^9$ 求所有生成树的权值和的 k 次方和

hihocoder Challenge 28

- $n, k \leq 50, e_w \leq 10^9$ 求所有生成树的权值和的 k 次方和
- author : skydec

hihocoder Challenge 28

- $n, k \leq 50, e_w \leq 10^9$ 求所有生成树的权值和的 k 次方和
- author：skydec
- k 次方和可以拆成选 k 条可以相同的边的乘积

hihocoder Challenge 28

- $n, k \leq 50, e_w \leq 10^9$ 求所有生成树的权值和的 k 次方和
- author：skydec
- k 次方和可以拆成选 k 条可以相同的边的乘积
- 直接暴力插值即可，$O(n^5)$

曼哈顿距离最大生成树

- $n \leq 10^5$

曼哈顿距离最大生成树

- $n \leq 10^5$
- $n \leq 10^6$

曼哈顿距离最大生成树

- $n \leq 10^5$
- $n \leq 10^6$
- easy

USACO OPEN17 grass

- 动态修改点的颜色，求不同色点间边权的最小值

USACO OPEN17 grass

- 动态修改点的颜色，求不同色点间边权的最小值
- $n \leq 5 * 10^5$

USACO OPEN17 grass

- 动态修改点的颜色，求不同色点间边权的最小值
- $n \leq 5 * 10^5$
- 两点间颜色不同，那么任何一个路径上都一定会有不同

USACO OPEN17 grass

- 动态修改点的颜色，求不同色点间边权的最小值
- $n \leq 5 * 10^5$
- 两点间颜色不同，那么任何一个路径上都一定会有不同
- 取最小生成树维护即可。

Matroid method

- 拟阵是图论里很有力的工具

Matroid method

- 拟阵是图论里很有力的工具
- 拟矩定义：独立集的继承性，独立集的增广性

Matroid method

- 拟阵是图论里很有力的工具
- 拟矩定义：独立集的继承性，独立集的增广性
- 经典的拟阵：(图，森林)，(向量，线性无关集)

Application1

- 来看一道金策的集训队胡策题。

Application1

- 来看一道金策的集训队胡策题。
- 维护一个高精度的二进制数的集合

Application1

- 来看一道金策的集训队胡策题。
- 维护一个高精度的二进制数的集合
- 支持区间 xor 上 w

Application1

- 来看一道金策的集训队胡策题。
- 维护一个高精度的二进制数的集合
- 支持区间 xor 上 w
- 区间修改为 w

Application1

- 来看一道金策的集训队胡策题。
- 维护一个高精度的二进制数的集合
- 支持区间 xor 上 w
- 区间修改为 w
- 询问全局的最大 xor 和

Application1

- 区间修改可以变成单点 xor 和删除

Application1

- 区间修改可以变成单点 xor 和删除
- 区间 xor？我们维护这个它的差分，自然就变成了单点修改

Application1

- 区间修改可以变成单点 xor 和删除
- 区间 xor？我们维护这个它的差分，自然就变成了单点修改
- 单点修改？可以变成单点删除和加入

Application1

- 区间修改可以变成单点 xor 和删除
- 区间 xor？我们维护这个它的差分，自然就变成了单点修改
- 单点修改？可以变成单点删除和加入
- 删除和加入维护线性基？不会删除？

Application1

- 考虑我们 LCT 维护最大生成树的经典做法

Application1

- 考虑我们 LCT 维护最大生成树的经典做法
- 他们都是拟阵，自然有相似的性质

Application1

- 考虑我们 LCT 维护最大生成树的经典做法
- 他们都是拟阵，自然有相似的性质
- 维护最大生成线性基即可，只要对每个元素维护他是哪些东西 xor 出来的

Application1

- 考虑我们 LCT 维护最大生成树的经典做法
- 他们都是拟阵，自然有相似的性质
- 维护最大生成线性基即可，只要对每个元素维护他是哪些东西 xor 出来的
- 然后加入一个元素的时候，从包含它的"环"里扔掉一个最小元素即可。

拟阵的交

- 最大最小定理：$\max_{I \in I_1 \cap I_2} |I| = \min_U r_1(U) + r_2(S \backslash U)$

拟阵的交

- 最大最小定理：$\max_{I \in I_1 \cap I_2} |I| = \min_U r_1(U) + r_2(S \backslash U)$
- 拟阵交算法：维护一个集合 $I \in I_1 \cap I_2$，每次顺着" 增广路"，试图拓展一个进入 I。

拟阵的交

- 最大最小定理：$\max_{I \in I_1 \cap I_2} |I| = \min_U r_1(U) + r_2(S \backslash U)$
- 拟阵交算法：维护一个集合 $I \in I_1 \cap I_2$，每次顺着" 增广路"，试图拓展一个进入 I。
- 把 $I + x \in I_1$ 的 x 加入 I 之后，把因为 I_2 的限制而不独立的环里某个 y 删去

拟阵的交

- 最大最小定理：$\max_{I \in I_1 \cap I_2} |I| = \min_U r_1(U) + r_2(S \setminus U)$
- 拟阵交算法：维护一个集合 $I \in I_1 \cap I_2$，每次顺着" 增广路"，试图拓展一个进入 I。
- 把 $I + x \in I_1$ 的 x 加入 I 之后，把因为 I_2 的限制而不独立的环里某个 y 删去
- 删去之后再加入满足 I_1 限制的 z，再删去 t....

拟阵的交

- 最大最小定理：$\max_{I \in I_1 \cap I_2} |I| = \min_U r_1(U) + r_2(S \backslash U)$
- 拟阵交算法：维护一个集合 $I \in I_1 \cap I_2$，每次顺着" 增广路"，试图拓展一个进入 I。
- 把 $I + x \in I_1$ 的 x 加入 I 之后，把因为 I_2 的限制而不独立的环里某个 y 删去
- 删去之后再加入满足 I_1 限制的 z，再删去 t....
- 直到某次某个 t 不用被删去了为止

矩阵的交

- 具体来说，假设加入 l 使其满足 l_1 约束的集合是 X_1

矩阵的交

- 具体来说，假设加入 I 使其满足 I_1 约束的集合是 X_1
- 加入 I 使其满足 I_2 约束的集合是 X_2

矩阵的交

- 具体来说，假设加入 l 使其满足 l_1 约束的集合是 X_1
- 加入 l 使其满足 l_2 约束的集合是 X_2
- 我们每次找一个从 X_1 到 X_2 的最短路增广

拟阵的并

- 两个拟阵的并的所有独立集，所有能分成两个部分分别为第一个拟阵和第二个拟阵的独立集的集合

拟阵的并

- 两个拟阵的并的所有独立集，所有能分成两个部分分别为第一个拟阵和第二个拟阵的独立集的集合
- 两个拟阵的并还是拟阵！

拟阵的并

- 两个拟阵的并的所有独立集，所有能分成两个部分分别为第一个拟阵和第二个拟阵的独立集的集合
- 两个拟阵的并还是拟阵！
- 令在第一个拟阵选一个独立集，并在第二个矩阵选一个独立集组成一个集合对的拟阵为 A

拟阵的并

- 两个拟阵的并的所有独立集，所有能分成两个部分分别为第一个拟阵和第二个拟阵的独立集的集合
- 两个拟阵的并还是拟阵！
- 令在第一个拟阵选一个独立集，并在第二个矩阵选一个独立集组成一个集合对的拟阵为 A
- 令在两个拟阵中不能重复选相同元素的拟阵为 B

拟阵的并

- 两个拟阵的并的所有独立集，所有能分成两个部分分别为第一个拟阵和第二个拟阵的独立集的集合
- 两个拟阵的并还是拟阵！
- 令在第一个拟阵选一个独立集，并在第二个矩阵选一个独立集组成一个集合对的拟阵为 A
- 令在两个拟阵中不能重复选相同元素的拟阵为 B
- 拟阵的交其实就是两个拟阵的并

拟阵的并

- 两个拟阵的并的所有独立集，所有能分成两个部分分别为第一个拟阵和第二个拟阵的独立集的集合
- 两个拟阵的并还是拟阵!
- 令在第一个拟阵选一个独立集，并在第二个矩阵选一个独立集组成一个集合对的拟阵为 A
- 令在两个拟阵中不能重复选相同元素的拟阵为 B
- 拟阵的交其实就是两个拟阵的并
- 带入最大最小定理直接得到
  $r(S) = \min_U r_1(U \cap S_1) + r_2(U \cap S_2) + |S\ U|$

Application2

- Alice 和 Bob 在玩游戏，给一个图，Alice 先手删去一条边，然后 Bob 固定一条边 <+-> 然后轮流操作，已经被固定的边不能再被删除，同理已经被删除的边不能再被固定。

Application2

- Alice 和 Bob 在玩游戏，给一个图，Alice 先手删去一条边，然后 Bob 固定一条边 <+-> 然后轮流操作，已经被固定的边不能再被删除，同理已经被删除的边不能再被固定。
- Bob 如果最后选中的边能给形成一棵生成树，那么 Bob 赢，否则 Alice 赢。

Application2

- Alice 和 Bob 在玩游戏，给一个图，Alice 先手删去一条边，然后 Bob 固定一条边 <+-> 然后轮流操作，已经被固定的边不能再被删除，同理已经被删除的边不能再被固定。
- Bob 如果最后选中的边能给形成一棵生成树，那么 Bob 赢，否则 Alice 赢。
- 问什么时候 Alice 有必胜策略，什么时候 Bob 有必胜策略。

Application2

- 一个图有两个不同的生成树，等于是图拟阵和自己的并有个大小为 $2(n-1)$ 的独立集

Application2

- 一个图有两个不同的生成树，等于是图拟阵和自己的并有个大小为 $2(n-1)$ 的独立集
- 等于是每个边集 U，都至少有 $m - |U| + 2r(U) \geq 2(n-1)$

Application2

- 一个图有两个不同的生成树，等于是图拟阵和自己的并有个大小为 $2(n-1)$ 的独立集
- 等于是每个边集 U，都至少有 $m - |U| + 2r(U) \geq 2(n-1)$
- 最小化左边，U 肯定是一个划分里的所有边，对于每个划分，如果他分成了 p 份

Application2

- 一个图有两个不同的生成树，等于是图拟阵和自己的并有个大小为 $2(n-1)$ 的独立集
- 等于是每个边集 U，都至少有 $m - |U| + 2r(U) \geq 2(n-1)$
- 最小化左边，U 肯定是一个划分里的所有边，对于每个划分，如果他分成了 p 份
- 因为 $(n-1) - r(U) = (n-1) - (n-p)$，其实就是说划分外的边大于等于 $2(p-1)$

Application2

- 一个图有两个不同的生成树，等于是图拟阵和自己的并有个大小为 $2(n-1)$ 的独立集
- 等于是每个边集 U，都至少有 $m-|U|+2r(U)\geq 2(n-1)$
- 最小化左边，U 肯定是一个划分里的所有边，对于每个划分，如果他分成了 p 份
- 因为 $(n-1)-r(U)=(n-1)-(n-p)$，其实就是说划分外的边大于等于 $2(p-1)$
- 如果没有两个不同的生成树所以存在一个划分只有 $2p-3$ 个边，Alice 只要先手干掉 $p-1$ 条边即可

Application2

- 反过来如果有两个不同的生成树

Application2

- 反过来如果有两个不同的生成树
- 无论 Alice 删除了啥，Bob 总能选对应的环上的边固定

Application2

- 反过来如果有两个不同的生成树
- 无论 Alice 删除了啥，Bob 总能选对应的环上的边固定
- 两个生成树都还是把这两个点缩起来之后的图里的生成树

Application2

- 反过来如果有两个不同的生成树
- 无论 Alice 删除了啥，Bob 总能选对应的环上的边固定
- 两个生成树都还是把这两个点缩起来之后的图里的生成树
- 判断是否有两个不同的生成树?

Application2

- 反过来如果有两个不同的生成树
- 无论 Alice 删除了啥，Bob 总能选对应的环上的边固定
- 两个生成树都还是把这两个点缩起来之后的图里的生成树
- 判断是否有两个不同的生成树?
- 拟阵交，请

NEERC16 G Game on Graph

- 人赢和妹子玩游戏，每人轮流在有向图上移动一步，人赢先手

NEERC16 G Game on Graph

- 人赢和妹子玩游戏，每人轮流在有向图上移动一步，人赢先手
- 妹子想要游戏无穷，有穷的话赢更好

NEERC16 G Game on Graph

- 人赢和妹子玩游戏，每人轮流在有向图上移动一步，人赢先手
- 妹子想要游戏无穷，有穷的话赢更好
- 人赢想要游戏有穷，有穷的话赢最好

NEERC16 G Game on Graph

- 人赢和妹子玩游戏，每人轮流在有向图上移动一步，人赢先手
- 妹子想要游戏无穷，有穷的话赢更好
- 人赢想要游戏有穷，有穷的话赢最好
- 求最后会怎么样，$n, m \leq 10^5$

NEERC16 G Game on Graph

- 把棋子位置和先手情况的 pair 作为一个点

NEERC16 G Game on Graph

- 把棋子位置和先手情况的 pair 作为一个点
- 对于妹子点后继点中有有穷点那么有穷

NEERC16 G Game on Graph

- 把棋子位置和先手情况的 pair 作为一个点
- 对于妹子点后继点中有有穷点那么有穷
- 对于人赢点后继点都是有穷点才会有穷

NEERC16 G Game on Graph

- 把棋子位置和先手情况的 pair 作为一个点
- 对于妹子点后继点中有有穷点那么有穷
- 对于人赢点后继点都是有穷点才会有穷
- 反向 bfs 标记

NEERC16 G Game on Graph

- 把棋子位置和先手情况的 pair 作为一个点
- 对于妹子点后继点中有有穷点那么有穷
- 对于人赢点后继点都是有穷点才会有穷
- 反向 bfs 标记
- 如果有穷的话胜负也是类似的，妹子要走到有穷中赢的点才会行

NEERC16 G Game on Graph

- 把棋子位置和先手情况的 pair 作为一个点
- 对于妹子点后继点中有有穷点那么有穷
- 对于人赢点后继点都是有穷点才会有穷
- 反向 bfs 标记
- 如果有穷的话胜负也是类似的，妹子要走到有穷中赢的点才会行
- 然后一样的反向标记一波

NEERC16 G Game on Graph

- 把棋子位置和先手情况的 pair 作为一个点
- 对于妹子点后继点中有有穷点那么有穷
- 对于人赢点后继点都是有穷点才会有穷
- 反向 bfs 标记
- 如果有穷的话胜负也是类似的，妹子要走到有穷中赢的点才会行
- 然后一样的反向标记一波
- 类似于 reachability game

????

- 类似的还有明天的考试题

????

- 类似的还有明天的考试题
- 祝大家好运