图论问题选讲

陈峻宇

Dec 20, 2023

All Pair Maximum Flow[1]

给你一个 n 个点 m 条边的图，它是平面上的正 n 边形和一些对角线，节点按逆时针方向编号为 1 到 n。对角线只可能在节点处相交。

每条边有一个容量 w。

设 $f(a,b)$ 表示 a 到 b 的最大流，求

$$\sum_{i=1}^{n}\sum_{j=i+1}^{n} f(i,j) \bmod 998244353$$

$$3 \le n \le 2\times 10^5, n \le m \le 4\times 10^5, 0 \le w \le 10^9$$

[1]2019 ICPC Asia-East Continent Final,problem K

做法

本题的做法如下：

每次找一条边权最小的边，满足它恰好有一侧是无界区域。将它删去，将它的边权加到它所在最小环的其他边上。可以证明这个操作前后任意两点的最大流大小不变。

最后得到一棵树，在这棵树上统计任意两个点的最大流即可。

复杂度 $O(n \log n)$。

正确性证明

我们需要证明原图在删除边前后任意两点的最大流 (最小割) 大小不变。设这条边为 e，考虑这条边所在的最小环 C。

性质一：原图上任意两点的最小割，只会割掉环 C 上 0 或 2 条边。且如果割掉 2 条边，一定可以有一条是 e。

证明：假设我们把原图的每个端点向外连一条射线，把无界区域分成 n 个部分，则对偶之后的图是一棵有 n 个叶子的树。而原图中的割对应着把叶子分成两个区间后，这两个区间之间的最短路。

而树上任意两个叶子之间的路径只会经过环 C 上 0 或 2 条边。

设 e 对应的叶子为 v，设环 C 上恰好有一侧是无界区域的边对应的叶子集合为 V。

1. 若最短路是 V 到 V，假设两个端点都不是 v，一定有一个端点和 v 在同一个区间，可以把它换成 v。
2. 若最短路是 V 内的点 x 到 V 外的点 y，假设 x, y 都不是 v。若 v 和 x 在同一个区间，则可以把 x 换成 v；若 v 和 y 在同一个区间，由于 e 边权最小，因此小于等于 y 对应边的权值，可以把 y 改成 v。

因此如果割掉 2 条边，一定可以有一条是 e。

正确性证明

下面证明删除边前后任意两点的最大流 (最小割) 大小不变。

先证明任意两点在原图的最小割 $\geq$ 在新图上的最小割。

设原图上最小割边集为 E，若 E 包含 C 上的 0 条边，则在新图上，E 也是一个大小相同的割；若 E 包含 C 上的两条边，其中一条是 e，则在新图上，$E \backslash \{e\}$ 也是一个大小相同的割。

再证明任意两点在新图的最小割 $\geq$ 在原图上的最小割。

设新图上最小割边集为 E，若 E 包含 C 上的 0 条边，则在原图上，E 也是一个大小相同的割；否则，在原图上，$E \cup \{e\}$ 是一个大小相同或者更小的割。

在删除 e 之后，原图可能形成若干个多边形，形成一个链结构。我们只需要保证对于任意一个多边形，它上面的任意两个点在后续操作中最小割大小不变，即可以分开考虑这些多边形。因此做法的正确性得证。

图论题[1]

给你一张 n 个点的竞赛图 (每两个点之间恰好有一条边的有向图)，对于每条边，输出翻转这条边的方向之后，整张图的强连通分量个数。

$1 \leq n \leq 5000$

[1] 2020 北大集训 D3T2

竞赛图的强连通分量构成了一个链的结构，翻转不同强连通分量之间边的情况很容易考虑。

只需要考虑每一个强连通分量。

性质：一个竞赛图如果是强连通的，那么存在哈密顿回路，且可以在 $O(|V|^2)$ 的时间内找到一个哈密顿回路。

证明：考虑数学归纳法。

1. 当 $n = 1$ 时，显然成立。
2. 假设 $n \leq k(k \geq 1)$ 时成立，对于任意一个 $k + 1$ 个点的图，我们任意取一个点 x。
   删去 x 后，将图分成若干个强连通分量，可以将这些强连通分量排成一个链状结构，使得前面的向后面的有连边。
   根据归纳假设，每个强连通分量都存在一条哈密顿回路，则这个图存在哈密顿路径。
   又因为这个图是强连通的，所以 x 一定向第一个强连通分量有连边，最后一个强连通分量一定向 x 有连边，于是我们找到了一条哈密顿回路。

翻转不在哈密顿回路上的边不会改变强连通分量的数量。

对于翻转在哈密顿回路上的边的情况，我们需要求的是删去这条边之后，哈密顿路径上有多少条边被反向边覆盖。

可以依次考虑这些边，问题转化为在序列上 $O(|V|^2)$ 次区间加，$O(|V|)$ 次查询区间上有多少个 0，差分后即可做到 $O(|V|^2)$。

Invertation in Tournament[1]

给你一个 n 个点的竞赛图。
每一次操作可以选择一个点 x，改变所有连接 x 的边的方向。
求至少经过多少次操作可以使这个图变成强连通图，以及操作方案数。

$3 \le n \le 2000$

[1]codeforces 1268 D

这个题需要用到竞赛图的一些性质。

性质一：一个竞赛图是强连通图等价于将出度序列排序，不存在长度为 $i(1 \le i < n)$ 的前缀，其和为 $\frac{i \cdot (i-1)}{2}$。

证明：

1. 如果存在这样的一个前缀，那么说明存在一个子图，它内部的点不向外连边，则图不是强连通的。
2. 对于任意一个非强连通竞赛图，前面强连通分量中每个点的度数一定大于后面强连通分量中每个点的度数，且对于最后一个强连通分量，它内部点的度数和为 $\frac{n_k \cdot (n_k-1)}{2}$，所以一定存在一个这样的前缀。

性质二：当 $n \geq 4$ 时，n 个点的强连通竞赛图一定有一个 $n-1$ 个点的导出子图，满足它也是强连通竞赛图。
证明：考虑图中任意一个点 x。

1. 去掉 x 后图仍然强连通，则满足条件。
2. 去掉 x 的图可以被划分成若干个强连通分量 $H_1, H_2, ..., H_k$，且对于 $1 \leq i < j \leq k$，H_i 中任何一个点到 H_j 中的任何一个点都有边。并且，H_1 中存在一个点 a 满足 x 到 a 有边，H_k 中存在一个点 b 满足 b 到 x 有边。
   如果 $\forall 1 \leq i \leq k, |H_i| = 1$，那么 $k = n-1 \geq 3$，我们任取一个 $i(1 < i < k)$，去掉 H_i 即可。
   否则，对于每个强连通分量 i，我们找一条哈密顿路径 $u_i \to v_i(u_i$ 可以等于 $v_i)(u_1 = a, v_k = b)$。
   设 $|H_j| > 1$。设 p 为 v_j 在哈密顿路径上的前一个点。则我们在图中删去 v_j，然后令 $v_j = p$，图中仍然存在哈密顿路径，于是加上 x 后我们得到了一条哈密顿回路。因此图是强连通的。

性质三：当 $n > 6$ 时，n 个点的竞赛图可以通过至多一次操作变成强连通图。

证明：

1. 存在一个大小 ≥ 4 的强连通分量。则由性质二，我们可以找到一个点 x，满足去掉 x 后这个强连通分量仍然强连通。又因为 x 到这个强连通分量中所有点的边方向一定不完全相同，所以我们对 x 进行一次操作后，这个强连通分量仍然是强连通的。而在操作后，可以发现，这个强连通分量现在可以到达所有强连通分量，也可以从所有强连通分量到达，于是整个图强连通。
2. 存在至少三个强连通分量，在中间的强连通分量中任意取一个点进行操作。操作后每个点都可以先走到最后一个强连通分量，然后通过中间这个点走到第一个强连通分量，进而到达所有点。

如果上面两个条件都不满足，则每个强连通分量都 ≤ 3，且只有两个强连通分量，于是总点数 ≤ 6。

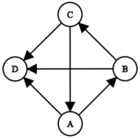
所以我们进行数据分治：

1. $n \leq 6$，枚举操作的点进行判断即可。
2. $n > 6$，首先判断原图是否强连通，如果不是，枚举每个点作为操作点，判断图是否强连通即可。具体实现可以用性质一中的结论。

复杂度 $O(n^2 \log n)$。

JYPnation[1]

给你一张 n 个点的竞赛图。图上不存在 4 个点 A, B, C, D，使得 $\{A, B, C, D\}$ 的导出子图为下图。



若从 i 出发可以到达 j，则定义 $\mathrm{dis}(i,j)$ 为从 i 到 j 经过的最小边数；否则定义 $\mathrm{dis}(i,j) = 614n$。求

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\mathrm{dis}(i,j)$$

$$4 \le n \le 8000, n \equiv 0 \pmod 4$$

[1] codeforces 1338E

性质

竞赛图没有上面的图案，会产生很多性质。

性质一：竞赛图如果有环，那么有三元环。

可以归纳证明。

性质二：若 i 能到达 j，则 $\mathrm{dis}(i,j) \le 3$。

因为如果 $\mathrm{dis}(i,j) \ge 4$，则存在一条路径 $p_1 \to p_2 \to p_3 \to p_4 \to p_5$ 满足不存在 $p_i \to p_j (i < j-1)$，则 (p_1, p_3, p_4, p_5) 形成了上面的图案。

性质三：对原图求强连通分量，会形成一条链，只有链上的最后一个强连通分量可能有多于一个点。

竞赛图不存在两个点的强连通分量。如果前面某个强连通分量的大小 ≥ 3，则由性质一，有三元环，会和后面强连通分量中的点构成上面的图案。

前面单点和其他点的答案很好计算，现在我们只考虑最后一个强连通分量。

性质

取入度最大的点 x，将点按照是否指向 x 划分成两部分，P 为指向 x 的点集与 x 的并集，Q 为 x 指向的点集。由于这是一个强连通图，所以 P, Q 均非空。

性质四：P, Q 均无环。

P 无环：如果 P 有环，这个环不包含 x。由性质一，有三元环，会和 x 构成上面的图案。

Q 无环：由于图强连通，所以 Q 中存在某个点指向 P 中不同于 x 的某个点，设为 y。

设 Q 中指向 y 的点集为 U，$Q \backslash U$ 为 V，则 U 非空。

U 中不存在环，否则有三元环，会和 y 构成上面的图案。

V 中的所有点指向 U 中的所有点，因为如果 $a \in U, b \in V, a \to b$，则 a, b, x, y 会形成上面的图案 $(y \to x)$。

V 无环，否则有三元环，会和 U 中的点形成上面的图案。

所以 Q 无环。

性质

将 P, Q 按照拓扑序排成 $p_1, p_2, \cdots, p_{n_1}$ 和 $q_1, q_2, \cdots, q_{n_2}$。
记 $\mathrm{inQ}(p_k)$ 为 Q 中指向 p_k 的点集。

性质五：$\mathrm{inQ}(p_k)$ 是 q 序列的一个后缀。
因为如果 $q_i \to p_k \to q_j (i < j)$，那么 p_k, x, q_i, q_j 形成了上面的图案。

性质

记 $\mathrm{inP}(q_k)$ 为 P 中指向 q_k 的点集。

性质六：$\mathrm{inP}(q_k)$ 是 p 序列的一个后缀。

如果 $p_i \to q_k \to p_j (i < j)$，

1. 若 $\mathrm{inQ}(p_i) \neq \emptyset$，设 $q_l \in \mathrm{inQ}(p_i)$。
   由 $\mathrm{inQ}(p_i)$ 是 q 序列的后缀，可得 $k < l, q_k \to q_l$。
   由 $\mathrm{inQ}(p_j)$ 是 q 序列的后缀，且 $q_k \to p_j$，可得 $q_l \to p_j$。
   此时 p_i, p_j, q_k, q_l 形成了上面的图案。
2. 若 $\mathrm{inQ}(p_i) = \emptyset$，则 $i \neq 1$，否则图不强连通。
   由 $\mathrm{inQ}(p_1) \neq \emptyset$，且是 q 序列的后缀，可得 $q_{n_2} \to p_1$。
   由 $\mathrm{inQ}(p_j)$ 是 q 序列的后缀，且 $q_k \to p_j$，可得 $q_{n_2} \to p_j$。
   此时 p_1, p_i, p_j, q_{n_2} 形成了上面的图案。

性质七：$|\operatorname{inQ}(p_k)| \geq |\operatorname{inQ}(p_{k+1})|$，且 $|\operatorname{inP}(q_k)| \geq |\operatorname{inP}(q_{k+1})|$。

可以由性质五和性质六推导出。

性质八：对于任意 p_i 和 q_j，有 $\operatorname{dis}(p_i, q_j) + \operatorname{dis}(q_j, p_i) = 3$。

1. $p_i \to q_j$ 若存在 $k < i$ 使得 $q_j \to p_k$，则 $\operatorname{dis}(q_j, p_i) = 2$。由性质七，$q_j \to p_l (l \leq k)$，因此 $q_j \to p_1$。所以如果 $\operatorname{dis}(q_j, p_i) \neq 2$，则有 $p_1 \to q_j$，由性质七，有 $p_k \to q_j (1 \leq k \leq n_1)$。这说明 q_j 的入度 $\geq n_1$，大于 x 的入度 $n_1 - 1$，与 x 是入度最大的点矛盾。
2. $q_j \to p_i$ 则 $p_i \to x \to q_j$，因此 $\operatorname{dis}(p_i, q_j) = 2$。

现在考虑计算答案

1. 点对 p_i, q_j 给答案的贡献是 3。
2. 若 $|\mathrm{inQ}(p_i)| = |\mathrm{inQ}(p_j)| (i < j)$，则 $\mathrm{dis}(p_j, p_i) = 3$，$p_i, p_j$ 给答案的贡献是 4。
3. 若 $|\mathrm{inQ}(p_i)| > |\mathrm{inQ}(p_j)| (i < j)$ 则 $\mathrm{dis}(p_j, p_i) = 2$，$p_i, p_j$ 给答案的贡献是 3。
4. 对于 q_i, q_j，和上面 p 是一样的。

复杂度 $O(n^2)$。

给你一个长度为 n 的非负整数序列 a_i。

有一张 $n+1$ 个节点的无向图 G，节点编号为 $0, 1, \cdots, n$。初始时图上没有边。

你可以进行任意多次操作，每次操作可以选择一个区间 $[l, r]$，花费 $\text{xor}_{i=l}^{r} a_i$ 的代价给 $l-1$ 和 r 连上一条边。

求使 G 连通的最小代价。

$1 \le n \le 10^5, 0 \le a_i \le 10^9$

令 $b_i = \operatorname{xor}_{j=1}^{i} a_j (i = 0, 1, \cdots, n)$，令 G' 为一张 $n+1$ 个点的无向完全图，节点编号为 $0, 1, \cdots, n$，连接节点 x 和 y 的边权值为 $b_x \operatorname{xor} b_y$。则原问题即为求 G' 的最小生成树边权和。

boruvka 算法是一个求最小生成树的算法。算法流程如下：

- 最开始图上没有边，每个点是一个连通块。
- 进行若干轮迭代。每一轮对于每一个连通块，找到一条连接这个连通块内部点和外部点的权值最小的边。依次考虑这些边，如果它连接的两个连通块还没有连通，就将它加入图中。这一轮完成后，一些连通块合并为更大的连通块。
- 对于一个连通图，每次迭代连通块个数至少减少一半，因此最多迭代 $O(\log |V|)$ 轮。

对于本题，在每一次迭代过程中，可以用 trie 树在 $O(|V| \log a)$ 的时间内对每个点找到不在它所在连通块内，且和它 xor 起来最小的点。

总复杂度 $O(n \log n \log a)$。

解法二：trie 树上贪心

对所有 b_i 建立 trie 树。

考虑 trie 树上的每个节点，如果它有两个子树，就从这两个子树内各挑出一个点，使得它们 xor 起来最小，将 xor 值加入答案。

正确性证明：在最优情况下，trie 树上一个节点子树中的所有点最多只能向外连一条边，且一定连在这个节点兄弟节点的子树内。

具体实现可以启发式合并，枚举点数较少的子树里的所有点，在另一棵子树上贪心查询。

总复杂度 $O(n \log n \log a)$。

新年的繁荣[1]

给你 n 个 0 到 2^m-1 之间的整数 $a_1, \cdots, a_n$。令 G 为一张 n 个点的无向完全图，其中第 i 个点到第 j 个点的边权为 $a_i \text{ and } a_j$。

求图 G 的最大生成树边权和。

$1 \le n \le 10^5, 0 \le m \le 18$

[1]UOJ 176

应用 boruvka 算法，我们需要在每一轮迭代中，对于每个点找到不在它所在连通块内，且和它 and 起来最小的点。

发现处理 xor 的 trie 树不再适用。

考虑子集和变换，求出两个长度为 2^m 的数组 l, r (下标从 0 到 2^m-1)，

$l[x] = \min\{k\text{所在连通块的编号}|k=1,2,\cdots,n\text{满足}x\text{是}a_k\text{的“子集”}\}$，

$r[x] = \max\{k\text{所在连通块的编号}|k=1,2,\cdots,n\text{满足}x\text{是}a_k\text{的“子集”}\}$。

那么对每个点，就可以从高往低一位位进行贪心。

总复杂度 $O((2^m+n)m\log n)$。

MST and Rectangles[1]

给你一个 $n \times n$ 的数组 a，初值为 0。

m 次操作, 每次给五个参数 x_1, y_1, x_2, y_2, w，对于满足 $x_1 \le i \le x_2, y_1 \le j \le y_2$ 的 i, j，给 $a_{i,j}$ 增加 w。

最后构造一个 n 个点的无向图 G，对于满足 $1 \le i < j \le n$ 的 i, j，在 G 中添加一条连接 i, j，边权为 $a_{i,j}$ 的边。

求 G 最小生成树的边权和。

$1 \le n, m \le 10^5, 1 \le x_1 \le x_2 < y_1 \le y_2 \le n, -10^6 \le w \le 10^6$

[1] CS Academy,round72

考虑 boruvka 算法，我们需要对每个连通块求出它连向块外的权值最小的边，只需要对每个点求出它连向它所在连通块外的权值最小的边。

为了方便处理，在给矩形 $[x_1, x_2] \times [y_1, y_2]$ 增加 w 时，我们同时也给 $[y_1, y_2] \times [x_1, x_2]$ 增加 w。

考虑从上往下进行扫描线，用线段树来维护权值最小值。

因为要限制连向所在连通块外，所以我们可以给线段树每个节点维护两个值，最小值和除了最小值所在连通块的最小值。

总复杂度 $O(n \log^2 n)$。

给你一个 n 个点的带权无向完全图。

固定一个根节点，对于一个生成树，每个点的代价是它到根的路径上的最小边权，这个生成树的代价是所有点的代价和。

求每个点作为根时的最小代价。

$1 \leq n \leq 2000$

首先可以发现一些性质。

性质一：至少有一个边权最小的边在最优解中。

否则将这条边加入，同时将不在这条边到根路径上的点都连向这条边的远离根的端点，答案一定更优。

性质二：最优解一定可以是由根开始的一条链。

如果一个节点有多个儿子，可以把其他儿子挂到一个儿子下面，答案不会变差。

记最小边权为 m，我们可以将所有边权都减去 m，最后给答案加上 $(n-1)\times m$，则此时最小边权为 0。

设这条链上的边由根到另一端边权为 $w_1, w_2, w_3, \cdots, w_{n-1}$，设第一条权值为 0 的边是 w_k。

性质三：w_1 到 w_{k-2} 是严格单调减的。

否则找到最小的 i 满足 $w_i \le w_{i+1} (i+1 \le k-2)$，可以将边 $i+1$ 和边 $i+2$ 断开，连接边 $i+1$ 左端点和 $i+2$ 右端点，把中间端点连到后面。

原来边 $i+1$ 右端点和边 $i+2$ 右端点的代价分别是 w_i 和 $\min(w_i, w_{i+2})$，之后代价至多 w_i，代价变小了。

那么如果 $w_{k-2} > w_{k-1}$，答案就是 $\sum_{i=1}^{k-1} w_i$，否则是 $2w_{k-2} + \sum_{i=1}^{k-3} w_i$。

发现他们都可以表示为 $2w_{l+1} + \sum_{i=1}^{l} w_i$(在第一个式子中 $l = k-1, w_{l+1} = 0$)。

将每个点初始距离设为这个点周围所有边的权值最小值的 2 倍，然后直接跑最短路就行了。

复杂度 $O(n^2)$。

CNNCT2[1]

给你两张 n 个点 m 条边的无向连通图，每张图中边的编号为 $1, 2, \cdots, m$。

求最少保留多少个编号，使得每张图在仅保留这些编号的边时都连通。

$1 \leq n, m \leq 300$

[1] codechef October Challenge 2019

设 D 为最小的编号集合使得两张图连通，设 $D_1 \subseteq D$ 满足 D_1 中的边构成图 1 的生成树，$D_2 \subseteq D$ 满足 D_2 中的边构成图 2 的生成树。

由 D 最小，可得 $D = D_1 \cup D_2$，则 $|D| = |D_1| + |D_2| - |D_1 \cap D_2|$，我们要最小化 $|D|$，只需最大化 $|D_1 \cap D_2|$，即求两张图各自生成树的最大编号交集大小，也即最大的编号集合，使得两张图中都没有环。

这是一个拟阵交问题。

拟阵交算法

给定两个拟阵 $M_1=(S,\mathcal{I}_1),M_2=(S,\mathcal{I}_2)$，和一个集合 $I\in\mathcal{I}_1\cap\mathcal{I}_2$，定义关于 I 的交换图 $D_{M_1,M_2}(I)$ 是一个二分图，两部分点集分别为 I 和 $S\backslash I$。从 $y\in I$ 指向 $x\in S\backslash I$ 的有向边存在当且仅当 $I-\{y\}+\{x\}\in\mathcal{I}_1$；从 $x\in S\backslash I$ 指向 $y\in I$ 的有向边存在当且仅当 $I-\{y\}+\{x\}\in\mathcal{I}_2$。
算法流程：令初始的 I 为 $\emptyset$，令 $X_1=\{x\notin I|I+\{x\}\in\mathcal{I}_1\},X_2=\{x\notin I|I+\{x\}\in\mathcal{I}_2\}$。每次在当前的交换图 $D_{M_1,M_2}(I)$ 中找到一条 X_1 到 X_2 的最短路径 P，令 I 变为 $I\Delta P$(对称差)。这样的一次过程称为增广过程，重复增广过程直到 X_1 不能到达 X_2 为止，此时 I 即为所求。

EGGFREE[1]

称一个有向图是没有蛋的，若它没有环，且对任意三个互不相同的点 x, y, z，若图中包含边 $x \to y$ 和 $x \to z$，那么图中包含边 $y \to z$ 或 $z \to y$。

给你一张 n 个点 m 条边的无向图，求一种将这张无向图的所有边定向的方案，使得最终得到的有向图是没有蛋的，或者确定不存在这样的方案。

$1 \le n, m \le 2 \times 10^5$

[1] codechef March Challenge 2020

经过观察可以发现，有解的必要条件是这张无向图是弦图，即任意一个 $k(k>3)$ 元环至少有一条弦。

接下来说明这张无向图是弦图也是有解的充分条件。首先考虑弦图的一些性质。

弦图——一些概念和性质

导出子图：对于图 $G=(V,E)$ 和 $V'\subset V$，令 $E'=\{(x,y)|(x,y)\in E\wedge x,y\in V'\}$，则 (V',E') 称为 G 关于 V' 的**导出子图**。

定理：弦图的任意导出子图也是弦图。

点割集：对于图 $G=(V,E)$ 上的两个点 x,y，若 $V'\subseteq V$ 满足 x,y 在 G 关于 $V\backslash V'$ 的导出子图中不连通，则称 V' 为 x,y 的**一个点割集**。若 V' 的任意子集都不是 x,y 的点割集，则称 V' 为 x,y 的**一个极小点割集**。

记 $N(x)$ 为图上与 x 相邻的点集。

定理：设 V' 是 x,y 的一个极小点割集，设 x,y 在 G 关于 $V\backslash V'$ 的导出子图中所在连通块分别为 V_x, V_y，则 $\forall v\in V'$，$N(v)$ 包含 V_x 和 V_y 中的点。

证明：假设 $v\in V'$，$N(v)$ 不包含 V_x 中的点或者不包含 V_y 中的点，则 $V'\backslash\{v\}$ 也是 x,y 的点割集，这与 V' 是 x,y 的极小点割集矛盾。

定理：弦图上任意两点 x, y 的任意极小点割集的导出子图是完全图。
证明：若极小点割集大小 ≤ 1，则满足条件。否则任取极小点割集上的两个点 u, v，由上定理，$N(u), N(v)$ 有 V_x, V_y 中的点。

设 u, v 在 V_x 内的最短路径为 $u - x_1 \sim x_2 - v$，在 V_y 内的最短路径为 $u - y_1 \sim y_2 - v$。考虑环 $u - x_1 \sim x_2 - v - y_2 \sim y_1 - u$，这是一个边数 ≥ 4 的环，环上存在一条弦。

由于 V_x, V_y 不连通，这条弦不能连接 V_x, V_y。由于最短路的性质，这条弦不能连接 V_x 或 V_y 内的两个点，也不能连接 u, v 与 V_x, V_y 中的点，那么只能连接 u, v。

因此这个极小点割集中每两个点都有边相连，导出子图为完全图。

单纯点：对于图 G 上的一个节点 x，若 $\{x\} \cup N(x)$ 的导出子图为完全图，则称 x 为 G 的一个**单纯点**。

定理：任何一个弦图都至少有一个单纯点，不是完全图的弦图至少有两个不相邻的单纯点。

证明：考虑数学归纳法，单独考虑每一个连通块。

首先，结论对所有完全图成立。

设一个连通弦图 $G = (V, E)$ 不是完全图，则 $|V| \geq 3$。设 x, y 满足 $(x, y) \notin E$。设 I 是 x, y 的一个极小点割集。设 x, y 在 G 关于 $V \backslash I$ 的导出子图中所在连通块分别为 V_x, V_y。

考虑 x 一侧的情况，设 $L = V_x \cup I$，设 G' 为 G 关于 L 的导出子图。若 G' 是完全图，则 $u = x$ 是 G' 的单纯点；否则，因为 G' 是一个点数更小的弦图，由归纳假设，有两个不相邻的单纯点，又因为 I 是一个完全图，所以 V_x 中一定有一个 G' 的单纯点 u。而 $N(u) \subseteq L$，因此 u 也是 G 的单纯点。同理 V_y 中有一个 G 的单纯点 v，且 u, v 不相邻。

弦图——一些概念和性质

完美消除序列：对于图 $G=(V,E), V=\{1,\cdots,n\}$，设 p 为 $1,2,\cdots,n$ 的一个排列，若 $\forall i=1,2,\cdots,n$，p_i 在 G 关于 $\{p_i,p_{i+1},\cdots,p_n\}$ 的导出子图中为单纯点，则称 p 为 G 的一个**完美消除序列**。
定理：一个无向图是弦图当且仅当它有完美消除序列。
证明：设一个无向图有完美消除序列，则任取图上的一个边数 ≥ 4 的环，设 x 为完美消除序列中第一个环上的点，设 x 在环上与 y,z 相连，则 $(y,z)\in E$，这个环上有弦。

设一个无向图是弦图，因为弦图有单纯点，所以可以通过数学归纳证明有完美消除序列。

弦图——最大势算法

　　最大势算法可以求出弦图的完美消除序列，可用于判定一个图是否是弦图。复杂度 $O(n+m)$。
算法流程：维护一个初始为空的序列，对每个点维护一个值，表示它周围有多少个点已经在序列中，每次选出一个值最大的点加入序列。最后将整个序列翻转。
　　如果这个图是弦图，最终得到的序列就是一个完美消除序列。设 $\alpha(x)$ 为 x 在这个序列中的位置。
正确性证明：我们需要证明对于任何一个弦图，算法求出的序列一定是一个完美消除序列，即在序列中位于点 x 后面的点两两相连。
引理一：考虑三个点 u, v, w 满足 $\alpha(u) < \alpha(v) < \alpha(w)$，如果 uw 相连，vw 不相连，则 w 只给 u 的值贡献，不给 v 贡献。为了让 v 比 u 先加入序列，需要一个 x 满足 $\alpha(v) < \alpha(x)$ 且 vx 相连，ux 不相连，即 x 只给 v 贡献而不给 u 贡献。

弦图——最大势算法

引理二：任意一个弦图一定不存在一个序列 $v_0, v_1, ..., v_k (k \geq 2)$ 满足下列性质：

1. $v_i v_{i+1} (i \in [0, k-1])$ 相连。
2. 任意 $v_i, v_j (|i-j| > 1)$ 不相连。
3. $\alpha(v_0) > \alpha(v_i)(i \in [1, k])$。
4. 存在 $i \in [1, k-1]$，满足 $\alpha(v_i) < \alpha(v_{i+1}) < ... < \alpha(v_k)$ 且 $\alpha(v_i) < \alpha(v_{i-1}) < ... < \alpha(v_1) < \alpha(v_k) < \alpha(v_0)$。

证明：由于 $\alpha(v_1) < \alpha(v_k) < \alpha(v_0)$，且 $v_1 v_0$ 相连，$v_k v_0$ 不相连，所以由性质一，存在 x 满足 $\alpha(v_k) < \alpha(x)$ 且 $v_k x$ 相连，$v_1 x$ 不相连。

考虑最小的 $j \in (1, k]$ 满足 $v_j x$ 相连，我们可以推出 $v_0 x$ 不相连，否则 $v_0 v_1 ... v_j x$ 构成了一个长度 ≥ 4 且无弦的环。

如果 $x < v_0$，则 $v_0, v_1, ..., v_j, x$ 也是一个满足性质的序列；如果 $v_0 < x$ 则 $x, v_j, ..., v_1, v_0$ 也是一个满足性质的序列。

在上面的推导中，我们扩大了 $\min(v_0, v_k)$，于是一直推下去，一定会产生矛盾。

弦图——最大势算法

定理一：对于任何一个弦图，最大势算法求出的序列一定是一个完美消除序列。
证明：考虑任意三个点 u, v, w 满足 $\alpha(u) < \alpha(v) < \alpha(w)$，我们需要证明若 uv 相连，uw 相连，则 vw 一定相连。考虑反证法，假设不相连，那么 w, u, v 就是一个满足引理二中性质的序列，我们证明了这样序列不存在，所以矛盾，vw 相连。

最后我们需要检验这个图是否是弦图，即检验最大势算法求出的序列是否是完美消除序列。

我们只需要检查任意三个点 u, v, w 满足 $\alpha(u) < \alpha(v) < \alpha(w)$，是否存在 uv 相连，uw 相连，而 vw 不相连的情况。

实际上并不用枚举所有的点，对于序列中 u 后面所有与 u 相连的点 $v_1, v_2, ..., v_k(\alpha(v_1) < \alpha(v_2) < ... < \alpha(v_k))$，只需要检验 v_1 与 $v_i(i = 2, \cdots, k)$，是否相连即可。

因为若其他点对不满足，考虑 x 最小的一对不满足点对 $v_x, v_y(2 \leq x < y \leq k)$，由于 $v_{x-1}v_x$，$v_{x-1}v_y$ 均相连，于是这一对点会在 v_{x-1} 处被检验出来。

回到原问题，我们需要说明这张无向图是弦图是有解的充分条件。

考虑完美消除序列，所有边定向为序列里靠前的点指向靠后的点。因为每个点 x 和与它相邻且位于它后面的点的导出子图是完全图，所以符合条件。

总复杂度 $O(n+m)$。

Data Center Drama[1]

给你一个 n 个点 m 条边的无向连通图，要求添加最少数量的边，然后给边定向，满足每个点的入度和出度都为偶数。求最小的添加边数量，以及任意一种定向方案。

$1 \le n \le 10^5, 1 \le m \le 2 \times 10^5$

[1] codeforces 528 C

因为每个点入度和出度都是偶数，所以在无向图中，每个点度数为偶数。设当前图中度数为奇数的点有 k 个，则至少要先加上 $\frac{k}{2}$ 条边，每条边连接一对度数为奇数的点。

因为每个点的出度都是偶数，所以图中的总边数应该为偶数。若此时图中有奇数条边，则至少要再加入一条边。为了保持每个点的度数为偶数，可以任意加一条自环。

在满足每个点的度数都是偶数，并且图中有偶数条边之后，我们一定能构造出合法的方案。因为此时图中存在一条长度为偶数的欧拉回路，只要让回路上相邻两条边的方向相反即可，这时，相邻两条边要么给公共点贡献两个出度，要么贡献两个入度。

复杂度 $O(n+m)$。

Mike and Fish[1]

平面上有 n 个格点，第 i 个点的坐标为 (x_i, y_i)。要求将每个点染成红色或者蓝色，使得每一行、每一列上的红色点与蓝色点个数之差的绝对值 ≤ 1。

$1 \leq n \leq 2 \times 10^5, 1 \leq x_i, y_i \leq 2 \times 10^5$

[1] codeforces 574 D

首先将行、列转化成点，每个点转化成边。

那么就有一个网络流的思路。设每个点一开始都是红色，我们要将一些点改成蓝色，对应在图上就是流这条边。然后限制可以转化为对每个点流量的限制，就可以用上下界网络流实现了。

实际上，由于这个限制很宽松，并不需要网络流。

仍然考虑上面的图，新建一个节点 0。如果一个点的度数为奇数，则我们让他向 0 连一条边。

那么现在每一个连通块中的每个点的度数都为偶数。

所以对每个连通块找一个欧拉回路，除了 0 所在的连通块以外，每个连通块的欧拉回路长度都是偶数 (因为这个图是二分图，所以总边数等于左边所有点的度数和，也为偶数)。若 0 所在的连通块边数为奇数，则让 0 向 0 连一条自环，这样 0 所在连通块的欧拉回路长度也为偶数。对于所有欧拉回路交替染色。这样，每个点连出的所有边中两种颜色的边数量相等，而原图所有点连出的边中两种颜色边的数量差也不超过 1。

复杂度 $O(n)$。

LLLGRAPH[1]

对于一个图 $G=(V,E)$，定义 $L(G)$，G 中的每一条边对应 $L(G)$ 中的一个点，若 G 中有两条边有共同的端点，则它们对应的点在 $L(G)$ 中相连。

给一个 n 个点 m 条边的图 G 以及一个数 k，求 $L^k(G)$ 的最大独立集大小是多少，答案对 998244353 取模。

$2 \le n, m \le 2000, 1 \le k \le 7$，$G$ 中无重边无自环。

[1] codechef April Challange 2020

当 $k=1$ 时，可以发现，我们需要求出 G 中的最大匹配，用带花树算法解决，复杂度 $O(mn\alpha(n))$。

当 $k\geq 2$ 时，我们需要找到 $L^{k-2}(G)$ 中的最大折线集合，使得每条边最多被用了一次。对于一个边数为 m 的连通块，答案就是 $\lfloor \frac{m}{2} \rfloor$。

证明：考虑对这个连通块求出 dfs 树，从下到上进行匹配。在点 i，考虑所有 i 连向儿子的还未匹配的边，i 连出的向下的返祖边，将这些边进行匹配，如果有剩下的，就和 i 连向 i 父亲的边进行匹配。这样只有 i 是根节点才可能剩下边。

可以发现，一个连通块中的边经过一次 L 变换之后是一个连通块。

因此我们需要对原图的每个连通块求出它经过 $k-2$ 次 L 变换的边数，即 $k-1$ 次变换后的点数。

当 $k=2$ 时，点数是 m。
当 $k=3$ 时，点数是

$$\sum_{i=1}^{n}\binom{d_i}{2}$$

当 $k=4$ 时，一个点对应了一个长度为 3 的链或者一个 3 叉形状，且链会产生一个点，3 叉形状会产生三个点，点数为

$$\sum_{i=1}^{m}(d_{x_i}-1)(d_{y_i}-1)+\sum_{i=1}^{n}3\binom{d_{x_i}}{3}$$

当 $k=5$ 时，记 $d1(x,y)=d_x+d_y-2, d2_x=\sum_{(x,y)\in E} d1(x,y)-1$
则点数为

$$\frac{\sum_{i=1}^{n} d2_i^2}{2}-\sum_{i=1}^{m}(d1(x_i,y_i)-1)^2+\sum_{i=1}^{m} 3\binom{d1(x_i,y_i)}{3}$$

因为每条边和自己相乘，被两个端点各计算了一次，所以系数是 -1。

当 $k=6$ 时，我们可以算出 $d2_{x_i,y_i}$，即可以算出第一项。

对于后面的两项，将 $d1$ 拆开，得到关于 d_x 和 d_y 的式子，如果只关于 d_x 或 d_y，则可以直接拆出来，如果是 d_x 与 d_y 的若干次幂之积，可以用类似 $k=4\to k=5$ 的方法计算 (非常恶心的式子)。

当 $k=7$ 时，对原图先进行一次 L 变换，即可用 $k=6$ 来求了。

复杂度 $O(n^2)$。