图论杂题选讲

热身题

在一个无限大的网格上，初始所有格子都是白色的，依次把 n 个格子涂黑。在每次涂完之后，判断是否满足黑色点组成了四连通块且白色点组成了八连通块。

$n \leqslant 5 \times 10^5, x, y \leqslant 10^9$

热身题

黑色格子较少，可以直接用并查集维护连通性。

对于白格子，容易发现整体八连通等价于所有与黑色格子八相邻的白格子八连通。

与黑色格子相邻的白格数量显然不超过 $8n$，可以按操作倒序用并查集维护。

热身题?

强制在线怎么做?

热身题?

强制在线怎么做?

黑格子还是可以用并查集正常维护，问题是怎么处理白格。

Tips: 平面图欧拉公式： $n-m+r=p+1$（其中 r 为封闭面数、p 为连通块数）

热身题!

以黑格子为点，相邻黑格的共边为边，转成对偶平面图。

手画一下可以发现，在黑色格子四连通的前提下：

原图白格八连通
等价于
新图中的封闭面只能对应于原图的黑色“田字”

套用欧拉公式进行检查即可。

CF19E Fairy

给定一张无向图，要求删去一条边使其成为二分图，问有几种删法

$n, m \leqslant 10^5.$

CF19E

图论题中有时候提出一棵生成树能让问题清晰很多，我们称一条非树边与其覆盖的路径是这条非树边的简单环。

找到原图的一棵生成树，一个比较经典的结论是，从经过次数奇偶性（或者说异或）的角度来讲，所有非树边对应的简单环事实上构成了原图所有环的一组基。
换句话说，**原图中的任何一个环都可以看成是由若干个非树边对应的简单环拼成的。**

接下来简单讨论即可，结论是删去的边必须被所有简单环是奇环的非树边覆盖，且不能被任何简单环是偶环的非树边覆盖。

CF888G Xor-MST

给定一张无向完全图，点 u 到点 v 的边权为 $a_u \oplus a_v$，求最小生成树的边权和。

$n \leqslant 2 \times 10^5, a_i < 2^{30}$.

CF888G

边权数量太多，把 a_i 挂到 Trie 上考虑。

可以发现，根据 Kruskal，要使选择的边 (u,v) 边权最小，其 lca 深度必然最大。

另一方面，初始时，对于所有最深的有两个子树的点，其两个子树内必然分别只有一个原图的节点，因此连边后只会构成一个连通块。

综合上面两个观察，由归纳法不难得出，在 Kruskal 的过程中，对每一个有两个子树的节点，其两个子树会形成两个连通块，而我们只需要在两个子树间连一条最小的边。

由于树高很小，暴力枚举其中一个子树的选择并在另一颗子树上查询即可，复杂度 $\mathrm{O}(n\log^2 a)$。

每次枚举小的子树可以优化到 $\mathrm{O}(n\log n\log a)$，对 Trie 建虚树+两棵子树同时遍历是可以做到 $\mathrm{O}(n\log a)$，但是都没必要。

P4151 最大异或和路径

给定一个带边权无向图，在所有从 1 到 n 的路径中（可以重复经过点或边），求经过的所有边的异或和的最大值。其中，如果一条边经过了多次，则异或多次。

$n, m \leqslant 10^5, w \leqslant 10^{18}$

P4151

注意到可以继续套用前面那题的思路。

还是提取出一个搜索树，可以发现任何一个路径都能表示为树上简单路径再并上若干简单环得到的组合。

因此把所有简单环的异或和丢进线性基里就做完了。

ARC084D Small Multiple

给定 K，问 K 的正整数倍中最小的数位和是多少。

$K \leqslant 10^5$.

ARC084D

(https://atcoder.jp/contests/arc084/tasks/arc084_b)
正难则反，枚举 K 的倍数并求最小和是困难的，这里我们考虑数位和尽量小的前提下，哪个数是 K 的倍数。

具体来说，考虑到到在十进制下可以用 $+1, \times 10$ 按位构造出所有数，我们可以构造如下的同余最短路图：
对 $1 \leqslant i < K$：

- $i \to [(i+1) \mod K]$，边权为 1
- $i \to [10i \mod K]$，边权为 0

显然对如上的图跑一遍起点为 1，终点为 0 的最短路（实际上可以用 $\mathrm{01bfs}$）即可。
对于进位的情况，数位和确实不是加一，但是显然这一情况也不是最短路，所以无影响。

P9140 背包

n 个有体积和价值的物品做完全背包，q 次询问，每次给你一个体积远大于单个物品的背包，求最大价值。

$n \leqslant 50, c_i \leqslant 10^6, v_i \leqslant 10^5$

$q \leqslant 10^5, 10^{11} \leqslant V \leqslant 10^{12}$

P9140

这么大的体积肯定不可能正常做完全背包。

考虑选出一个基准物品，设其体积为 m，价值为 c，利用模 m 同余最短路做背包。

这里与之前不同的地方在于，不同方案的“性价比”可能不一样了，凑出占用体积最小的方案再用剩下空间全选 m 并不一定最优。

考虑对于一个占用了体积 V' ，总价值为 C 的合法方案，我们最终的价值为 $C + \frac{V-V'}{m} \times c$。

因此相比于原来我们记录达到每个同余系的最小体积，我们可以转为记录达到某一同余系的最大权值 $C - \frac{V'}{m} \times c$。具体来说，由于现在 $\frac{V'}{m}$ 不一定是整数，可以在每次模 m 进位的时候减 c。

这样就做完了吗?

P9140

并没有。事实上这样会带来几个新问题：如何确保这样的图中最长路存在（没有正环）？如何确保最长路对应方案的 $V' \leqslant V$？

对于问题一，考虑一个正环的实际含义：这代表存在一个体积是 m 的整数倍的方案，它的性价比高于同体积的基准物品，因此以 $\frac{V'}{m} \times c$ 的代价去置换 V' 体积的基准物品是能让价值增加的。为了避免这一情况，自然可以想到选取性价比（$\frac{c}{m}$）最高的物品作为基准物品。
事实上这一改进也符合我们的直觉：在背包空间远大于物品时，应当先把剩余空间凑成 m 的倍数，再全部塞进性价比最高的物品。

P9140

对于问题二，根据最长路的性质，可以知道每个点最多经过一次，因此对应实际方案的占用体积不超过 $m \times \max\{v\} \leqslant 10^{10} < V$，因此题目性质保证了这一情况不会发生。

至此我们得到了一个 $\Theta(E \log E)$ （Dijkstra）或者最坏 $\mathrm{O}(VE)$ (SPFA) 的正确算法，其中 $V \sim n, E \sim nm$。因此 Dijkstra基本跑不过去，SPFA由于图的特殊性反而可以通过。

但是实际上有一个更快且更好写的做法：由于本质上是在做模意义下的完全背包，因此转移顺序是任意的，因此我们可以钦定按顺序分别对每个物品计算转移。
而每个物品会在图中构成 $\gcd(v_i, m)$ 个子环，由于最短路不经过重复的节点，每个物品不可能在环上贡献一圈，因此只需要对每个子环转两圈即可完成更新。

对每个物品，处理所有子环的复杂度是 $\mathrm{O}(m)$，因此总复杂度为 $\mathrm{O}(nm)$，可以通过而且代码实现非常简单。

P3825 游戏

构造一个 n 位字符串，字符集为 $\{\mathrm{a}, \mathrm{b}, \mathrm{c}\}$，且第 i 位的字符不能为 c_i。

特别地，有 d 个位置没有上述限制，给出任意构造或者判断无解。

同时，还有 m 条额外的限制，每一条形如：如果第 p 个字符为 x，则第 q 个字符必须为 y。

$d \leqslant 8, \quad n, m \leqslant 10^5$

P3825

如果 $d=0$，则是经典 2 - SAT 问题。

对于 $d\neq 0$，已知 3 - SAT 是 NP 的。但由于 d 很小，容易想到暴力 3^8 枚举对应位置，复杂度 $\mathrm{O}(3^d(n+m))$，不足以通过。

一个想法是，由于我们每次枚举后都是通过 2 - SAT 求解，因此枚举的条件实际上太强了。

我们转而枚举每个位置**不是**什么字符，由于 “不是 a” 和 “不是 b” 实际上就覆盖了三种情况，因此复杂度降为 $\mathrm{O}(2^d(n+m))$，可以通过。

CF1515G Phoenix and Odometers

给定一个带边权有向图，q 次询问，每次给出 u, s, t，求是否存在一个经过 u 的回路（可重复经过点和边），使其边权和模 t 等于 s

$n, m, q \leqslant 2 \times 10^5$，$w, s, t \leqslant 10^9$

CF1515G

显然，一个环必然在同一个强连通分量内。由于将一个环走 t 次相当于没走，而分量内任意两点相互可达，所以一整个强连通分量对于同一对 (s, t) 的答案应该是相同的。下面对一个强连通分量单独考虑。

注意到，模意义下，实际上与异或类似，我们也可以随意加上或者减去（走 $t-1$ 次）一个环。因此一个自然的想法是像前面一样，提取出这个分量的一组基，使得分量内所有的环都可以由基中的环线性组合而成。

设这组基的 $\gcd$ 为 g，由裴蜀定理， 我们可以构造出任意的 g 的倍数的环（负数解对应减去），因此，只需再用一次裴蜀定理检查 $ug + vt = s$ 是否有解即可得到答案。

问题变为如何找到所有环的一组基。

CF1515G

注意到，图是强连通的，因此如果存在一条 u 到 v 长度为 l 的路径，那么模意义下我们必然可以构造出一条从 v 到 u 长度为 $-l$ 的路径。

换句话说，从路径长度的角度，我们其实可以将边看作双向的。因此如果设搜索树中根 r 到点 u 的树边距离为 $p(u)$，那么我们可以继续套用原来无向图的结论：非树边 $e=(u,v,w)$ 对应简单环 $p(u)+e-p(v)$（换句话说，$r \to u, u \to v, v \to r$），而所有简单环构成一组基底。

因此我们只需要对所有简单环长 $|p(u)|+w-|p(v)|$ 求 $\gcd$ 即可。

CF1458D Flip and Reverse

给定一 01 串，每次可以选择一个包含等量 $0, 1$ 的子串，将这一子串先取反再左右翻转，求任意次操作后能得到的最小字典序。

$n \leqslant 5 \times 10^5.$

CF1458D

看到 01 数相等，肯定先将 0 变 -1 转前缀和序列 s。
手玩一下，可以发现一次操作相当于选择一个 $s_{l-1}=s_r$ 的区间 $[l,r]$，把 s_l 到 s_r 左右翻转。

由于区间端点的选择只与端点处的值有关，而与位置无关，因此可以尝试把 s_i 值相同相同的位置的信息记录到一起。例如，最后可能会对每个固定的前缀和值得到一张表：前缀和第一次变成 x 时，下一个位置是 $x+1$，第二次是 $x-1$，……

不难发现这事实上构成了一个以 s_i 为顶点、s_i 向 s_{i+1} 连边的有向图，并且每个结点的出边有确定的顺序。在这张图上，一次操作相当于把一个环反向，而题目要求的是起点是 0 的字典序最小的欧拉路径。

事实上，利用调整法，容易证明即使不考虑边的顺序，原图的任意一条欧拉路径也都是合法的。因此只需要对原图贪心地求字典序最小的欧拉路径即可。

CF1163F Indecisive Taxi Fee

给定一张带边权无向图，q 次询问，每次求在原图中将一条边的边权改为 x 后 1 到 n 的最短路（不同修改间互不影响）。

$n, m, q \leqslant 2 \times 10^5,\ \ w, x > 0.$

CF1163F

可以将询问分为四类：修改的边是/否在原图的最短路上，边权比原来变大/变小。
可以发现，除了将最短路上的边变大以外的情况都是平凡的，且经过新边的最短路也是平凡的，下面只考虑将最短路上的边变大且强制不经过这一条边的最短路怎么求。

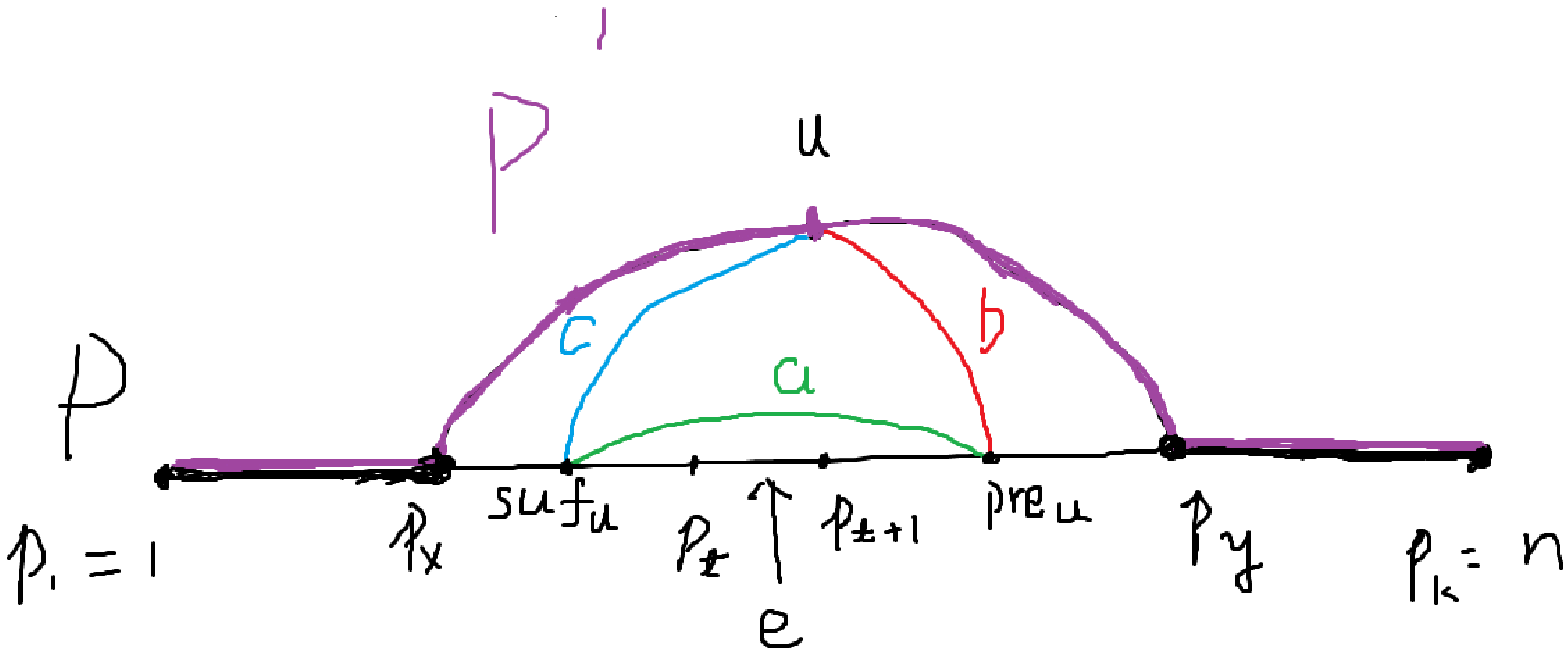
固定原图的一条最短路序列 P: $p_0 = 1, p_1, \cdots p_{k-1}, p_k = n$，对于不经过 P 上给定边 $e = (p_t, p_{t+1})$ 的最短路 P'，容易证明其必然有一段前缀与 P 重合，一段后缀与 P 重合，且剩余部分完全不重合。换句话说，可以设
$P' = (p_1, p_2, \cdots, p_x, q_1, \cdots, q_r, p_y, p_{y+1}, \cdots, p_k)$，其中任意 $q_i \notin P$.

同理，对于原图中任意一点 u，1 到 u 的最短路必然与 P 恰好只有一段前缀重合，记其最后一个重合点下标为 pre_u，后缀类似地记为 suf_u。有结论：

CF1163F

- P' 中不存在点 u，使得 $pre_u \neq x, suf_u \neq y.$

证明：设存在 u，则必然有 $pre_u > t, suf_u < t+1$，如图，依据最短路的性质，同时有 $a+b \leqslant c$、$a+c \leqslant b$，又图中边权全部为正，矛盾！



CF1163F

> P' 中不存在点 u，使得 $pre_u \neq x, suf_u \neq y$.

因为 $pre_{p_x} = x, suf_{p_y} = y$，利用上述结论立即可得：

- 必然存在一条边 $(u, v) \in P'$，满足 $pre_u = x, suf_v = y. (*)$

又由于有 $x \leqslant t, y \geqslant t + 1$，因此此时 $|P'|$ 应该恰好等于 $dis(1, u) + w + dis(v, n)$.

CF1163F

> 必然存在一条边 $(u,v)\in P'$，满足 $pre_u=x,suf_v=y.$ $(*)$

注意到 pre,suf 数组与询问给出的边 e 完全无关，因此我们可以对每条边 (u,v)（有向），预处理出过该边的最短路 $dis(1,u)+w+dis(v,n)$，并将其记在 (pre_u,suf_v) 的位置。

而对于一个询问 (p_t,p_{t+1})，根据性质 $(*)$，只需用一些数据结构查询所有 $x\leqslant t,y\geqslant t+1$ 的位置 (x,y) 处的最小值即可（例如，set 或线段树）。

复杂度 $\mathrm{O}(m\log m+q)$.