图论知识点

> 天色刚刚破晓 紧箍还没做好
>
> 我还能微笑

东西还是不少的（但是省选前笔者定的就是先复习图论，没办法了！）。

整体上比较基础，主要是用作复习，都是正式比赛可能会出现的（基本只有模拟赛可能出现的会提及，但不会讲）。

基本概念

用 $G=(V,E)$ 表示图。$d(v)$ 代表节点 v 的度数，如果 $d(v)=|V|-1$，则称 v 为**支配点**。如果每个点的度数都是 k，则该图为 $k-$**正则图**。

将图用最少颜色数染色，使得同色点互不相邻。该数量称为**图的色数**。

经过不重复边的路径叫做**迹**。

子图

对一张图 $G=(V,E)$，若存在另一张图 $H=(V',E')$ 满足 $V'\subseteq V$ 且 $E'\subseteq E$，则称 H 是 G 的**子图**，记作 $H\subseteq G$。

若对 $H\subseteq G$，满足 $\forall u,v\in V'$，只要 $(u,v)\in E$，均有 $(u,v)\in E'$，则称 H 是 G 的**导出子图**。点集为 $V'(V'\subseteq V)$ 的导出子图称为 V' 导出的子图，记作 $G[V']$。

若 $H\subseteq G$ 满足 $V'=V$，则称 H 为 G 的**生成子图/支撑子图**。

如果一张无向图 G 的某个生成子图 F 为 $k-$正则图，则称 F 为 G 的一个 $k-$**因子**。

如果有向图 $G=(V,E)$ 的导出子图 $H=G[V^*]$ 满足 $\forall v\in V^*,(v,u)\in E$，有 $u\in V^*$，则称 H 为 G 的一个**闭合子图**。也就是说，图内部是闭合的，不存在一个点在导出子图内，可以通过原图的一条边连到一个不在导出子图内的点。

特殊的图

对于无向简单图，所有本来在图上的边都不在，本来不在的都在，那么这个图就是原无向图的补图。

对于有向图，每条边的方向取反，得到的图就是原图的反图。

特殊集合

一些特殊的点和边的集合有着特殊的意义，这里我们介绍一些常见的。

支配集

对于无向图，如果一个点集的点可以连接到原图的所有点，那么这个点集为原图的**支配集**。

这个问题是 NPH 的，我们通常使用 $O(2^n)$ 的枚举算法来求解支配集。

独立集

就是任意两点不相邻的点集。对于树和二分图我们有高效做法，但是一般图上，这个问题是 NPH 的。

匹配

对于图 $G=(V,E)$，若 $E'\in E$ 且 E' 中任意两条边都没有公共端点，且 E' 中没有自环，那么 E' 是 G 的一个**匹配**，也称为**边独立集**。如果一个点被匹配的边连接了，那么它就是**被匹配的**，否则就是**不被匹配的**。

边数最多的称为**最大匹配**，如果边带权，那么权重之和最大的匹配称为图的**最大权匹配**。

如果一个匹配在加入任何一条边后都不再是一个匹配，那么这个匹配就是**极大匹配**，最大匹配一定是极大匹配。

如果所有点都被匹配了，那么这个匹配是**完美匹配**。如果在一个匹配中只有一个点不被匹配，那么该匹配为**准完美匹配**。

对于一个匹配 M，若一条路径以非匹配点为起点，每相邻两条边中的一条在匹配中而另一条不在匹配中，那么这条路径称为**交替路径**；一条非匹配点终止的交替路径称为**增广路径**。

点覆盖

如果所有边都至少有一个端点在这个点集中，那么这个点集被称为**点覆盖集**。

点覆盖集一定是支配集，但是极小点覆盖集不一定是极小支配集（考虑一个三元环）。

点覆盖集拥有以下性质：

- 一个点集是点覆盖的充要条件是其补集是独立集。
- 一张图的任何一个匹配的大小都不超过其任何一个点覆盖的大小。

边覆盖

当前边集满足任何一个点都至少是其中一条边的一个端点，那么这个边集称为**边覆盖集**。

如果知道了最大匹配，那么将所有非匹配点都连一条边，那么就得到了一个最小边覆盖。同理，如果知道了最小边覆盖，那么将有公共点的边删去一条就得到了最大匹配。

团

一个图的子点集 V' 中任意两个不同的顶点都相邻，则称 V' 是图 G 的一个**团**。团对应的导出子图是完全图。说白了最大团就是最大完全子图。

求解一个图的最大团是 NPH 的，可以使用最大团搜索算法（在暴力枚举的基础上加一个不可能成为答案的最优性剪枝）来解决规模较小的图的问题。

拓扑排序

对于一个有向无环图（DAG）G，将 G 中所有顶点排成一个线性序列，使得图中任意一对顶点 u 和 v，若它们之间存在一条有向边 (u,v)，则 u 在线性序列中出现在 v 之前。

如果要让越小的 i 尽可能地出现地早，那么就是让最大的尽可能晚地出现，那么要求**反图上字典序最大的拓扑序**，也就是菜肴制作。

Erdős–Gallai 定理

令 $S=(d_1,\cdots,d_n)$ 代表简单无向图的度数，而且 d 为**非递增序列**，则无向图存在当且仅当 $\sum d$ 是偶数，而且 $\forall k$, i.e.:

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

> [CF1091E] New Year and the Acquaintance Estimation.
>
> 一张 $n+1$ 个点的无向图，给定前 n 个点的度数，问 $n+1$ 号点可能的度数。$n\le 5\times 10^5$。

二分出可能的度数，然后 Erdős–Gallai 定理不难在预处理前缀后做到 $O(n)$ check。代码。

欧拉路径

- **欧拉回路**：通过图中的每条边恰好一次的回路；
- **欧拉通路**：通过图中的每条边恰好一次的通路；
- **欧拉图**：有欧拉回路；
- **半欧拉图**：有欧拉通路但是没有欧拉回路。

对于无向连通图，如果所有点的度数均为偶数，那么是欧拉图；如果有恰好两个奇度数点，那么是半欧拉图。

而对于一张有向图（显然，它至少需要弱连通），是欧拉图当且仅当其是一个强连通图且每个节点的入度和出度相等。如果这张图**恰存在**一个顶点的出度比入度小 1，另一个点出度比入度大 1（这个点为起点），这个图存在欧拉通路。

求解欧拉路可以使用 Hierholzer 算法。采用 DFS 不断找环，遍历当前节点 u 的所有出边，如果没有走过那就遍历，遍历完所有出边后将 u 加入欧拉路径，最后如果遍历的点的个数为 $m+1$，那么就得到了反着的欧拉路径，否则欧拉路径不存在。

在找欧拉回路时，可以从任意节点出发。否则，需要从根据性质找到的点出发。代码。

哈密顿路径

将欧拉路边的相关定义换成点就成了哈密顿路。通过图中所有顶点一次且仅一次的通路称为哈密顿通路。通过图中所有顶点一次且仅一次的回路称为哈密顿回路。具有哈密顿回路的图称为哈密顿图。具有哈密顿通路而不具有哈密顿回路的图称为半哈密顿图。

不同于欧拉路，哈密顿路问题不存在多项式复杂度算法。人们尝试过许多方法，包括尝试转化成欧拉路，拆点限制只经过一次然后转化为网络流等，但很可惜都是假做法。

不过网络流的这个做法真的很有启发性，有些特殊条件的图真的可以使用它来求解。

Ore 定理。对于一个简单无向图，如果任意两个**不相邻**点度数和大于等于 n，那么这个图存在哈密顿回路，大于等于 $n-1$ 时存在哈密顿通路。这是一个充分不必要条件。其构造性证明可以在搜索 SGU122 的题解。

最短路问题

对于无权图（01 带权），可以使用 BFS 求解最短路。

对于多源最短路，可以使用 Floyd 算法，也就是通过 n 轮 DP 来求解最短路。

对于单源最短路，可以使用 Dijkstra 和 SPFA。Dijkstra 基于贪心的思想，每次寻找当前最短的路来走，正确性基于边权非负；SPFA 则通过 $O(nm)$ 的迭代来更新最短路，进而可以判断负环的存在性。

Johnson 通过将边改造为 $(u,v,w+d_u-d_v)$ 来实现，边权是三角形不等式来满足 ≥ 0，建立超级源点跑 SPFA 即可。

跑一遍最短路得出的树叫做最短路树。

如果我们要求的是最短简单路径（有负环），那么它是一个 NPC 问题。

差分约束

根据三角形不等式进行连边，然后用 SPFA 判断负环。

值得注意的是，如果使用 SPFA 求最短路，那么得到的是字典序最大的解。对于字典序最小的解，只需要将约束条件统统变为 $x_i - x_j \geq y$，然后跑最长路，有正环时无解（就是边的方向和权值都取反）。

[省选联考 2021 A 卷] 矩阵游戏。答案的构造是容易的，然后需要调整这个答案，让每一行和列依次 $+1, -1$，然后错开，直接差分约束即可。代码。

斯坦纳树

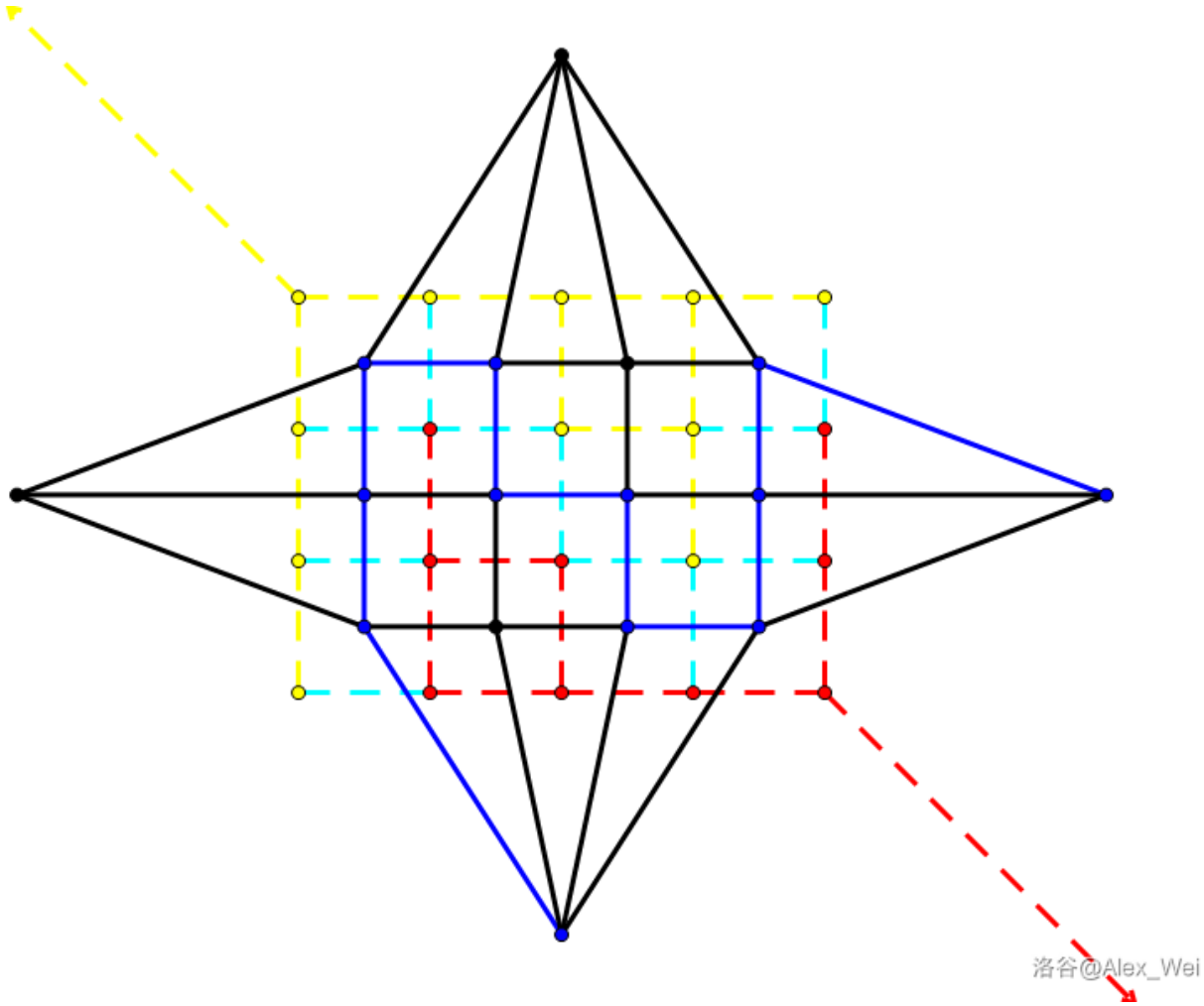
如果给定 n 个点，试求连接此 n 个点，总长最短的直线段连接系统，并且并且任意两点都可以通过系统中的直线段组成的折线连接起来。这是一个 NPH 问题。

模板。设 $f(i, S)$ 表示以 i 为根的一棵树，包含集合 S 中所有点的最小边权值和。有转移：$f(i, S) \leftarrow \min\{f(i, T) + f(i, S - T)\}$, $f(i, S) \leftarrow \min\{f(j, S) + w(i, j)\}$。前者可以使用子集 DP 实现，后者可以跑一个最短路（由于图很难特殊构造而且规模很小，所以实际上更建议 SPFA），时间复杂度是 $O(n \times 3^k + nm2^k)$。

平面图最小割

如果图 G 能画在平面 S 上，即除顶点处外无边相交，则称 G 可平面嵌入 S，G 为可平面图或平面图。画出的没有边相交的图称为 G 的平面表示或平面嵌入。

平面图可以转为对偶图，对偶图的最短路等于原平面图的最小割。给 G 的每个面搞一个点，两个面的公共边可以确定一条与其方向垂直的边。给源点和汇点连线可以将原图分成两个部分，跑最短路即可。



注意左侧和下侧，上侧和右侧分别是同一个点，从右上到左下的最短路即为左上到右下的最小割。

同余最短路

模板。这是一个最短路的变式问题。可以用于求解在某个范围内有多少重量可以由若干物品的完全背包凑出，就是多少数值可以由一些给定的数 b_i 由 $\sum a_i b_i (a_i \geq 0)$ 得到。

我们可以发现，如果 x 可以被表示出，那么 $x + ka_i (k > 0)$ 就可以被表示出。因此我们找一个最小的 a_1，然后连 $j \to (j + a_i) \bmod a_1$ 的长度为 a_i 的边，然后我们从 0 开始跑最短路。由于这里图的形态不太能特殊构造，因此使用 SPFA 往往会跑的更快。最后求出的 f_i 代表最小的能被凑出的数，满足 $f_i \bmod a_1 = i$。代码。

答案的求解十分容易。$[0, r]$ 的答案数量为：

$$\sum_{i=0}^{a_1-1} \max\left\{0, \left\lfloor \frac{r - f_i}{a_1} \right\rfloor + 1\right\}$$

但为什么要使用最短路呢？实际上这东西是体积模 m 意义下的完全背包，如果重复经过一个点，那么可以选择 $\frac{m}{\gcd(v_i, m)} - 1$ 个这类物品。也就是说，会在大小为 m 的环上形成 $\gcd(v_i, m)$ 个子环。

那么在每个子环上转两圈即可统计到所有转移，时间复杂度 $O(nm)$。代码。

[THUPC 2023] 背包。

如果我们将密度最大的物品选做基准物品，那么其它物品的选择可以替换为若干基准物品，这样可以最大化贡献。设基准物品体积为 w，贡献为 m。

设 f_i 代表最大的贡献，满足 $V \bmod m = i$。最终权值为 $f_i + \frac{V-V'}{m}w$，因此要最大化 $V - \frac{V'}{m}$，因此贡献应该是 $f_p + c_i - \frac{p+v_i}{m}w$。

可以发现每个 f_i 对应的物品个数一定是不超过 v 的，因此这一部分总容积不超过 $v^2 \le V$，不存在误判成有解的情况。代码。

删边最短路

> CF1163F.

首先一个基本事实：强制不走一条边 (u, v) 的最短路一定只走一条非最短路树上的边。否则之前求出的最短路一定是假的。

求出 $1, n$ 的最短路树 T_1, T_n。如果改的边不是最短路上的边的答案是好算的，否则，我们要算出强制不经过一条边的新的最短路。

我们需要保证 T_1, T_n 上 $1 \sim n$ 的最短路是相同的一条，否则无法计算。

求出 $p1_i$ 代表 $T_1(1 \to i)$ 与更新路径的最后一个交点（在最短路树上跳，第一个到的最短路上的点，就是 i 与 n 的 LCA），pn_i 同理。

这样维护先修改再单点查询的区间 `ckmin` 即可。代码。

显然，这种做法并不能在有向图上成立，因为不在最短路上的边可能不止一条。但是，走的路径依然满足中间只有一段不在最短路上的路径。

对于随机有向图可以采用这样一种方式处理：按照顺序遍历原最短路上的边，然后在起点上跑 SPFA，开个堆维护从哪里开始剩下的都走最短路。

k 短路

先建出一棵以 t 为根的最短路树 T，x 到 t 的最短路径为 d_x。设 $s \to t$ 的路径上不在 T 中的当前选择的路径的边集为 P'，$s \to t$ 上的所有边为 P，那么满足：

1. 将一条边 e 的代价定义为 $\Delta e = w - (d_u - d_v)$，那么 $L_{P'} = d_s + \sum_{e \in P'} \Delta e$；
2. 将 P 和 P' 中所有边按照 $s \to t$ 经过的顺序依次排列，那么对于 P' 中相邻的边 e_1, e_2，那么 $v_{e_1} = u_{e_2}$ 或者 u_{e_2} 是 v_{e_1} 在 T 上的祖先。
3. 对于每一个合法的 P'，有且仅有一个 P 与之对应。因为可以根据 P' 还原在 T 上选择了什么。

也就是说，我们现在要求满足性质 2 的第 k 小 L_p。

我们记录最后一条边和当前 L_p 的值即可表示 P'。初始我们将 1 所有在 T 上的祖先的所有的边中 Δe 最小的一条边加入小根堆，然后扩展时只有两种选择：

1. 删掉 P' 结尾的那条边，换成第二大的边；
2. 从 P' 的结尾开始到 T 的路径上，选择最小的边加入。

已知我们开始的描述路径的方式是不漏的，而且我们相当于枚举了所有的待替换边是否进行替换，因此这么做是正确的。

时间复杂度 $O(m\log m+k\log k)$，模板，代码。

生成树问题

可以使用以下方法求解：

- Kruskal：基于贪心的思想，按照边权从小到大排序；
- Prim：基于贪心的思想，每次找到不在最小生成树集合，维护 d_i 代表与当前树种权值最小边的权值。

对于次小生成树，其与与最小生成树最多仅有一条边的差距，枚举不是 MST 上的边，考虑删去树上的一条最小边，然后树上倍增找最大值即可。

DFS 或者 BFS 也能构建一棵生成树。对于有些题，我们会根据条件构建一棵生成树（或者是随便一棵生成树），然后再去进行操作。

一些性质

在一张图的所有 MST 上，一个权值的边的数量是一定的。

> [CF891C] Envy.
>
> 给定一个无向图，每次询问给定一些边的编号，问这些边是否能同时出现在一棵 MST 上。

如果这个询问的每一条边分别都能出现在 MST 上，那么这个询问就是合法的。

离线，按照边权进行排序。对于一条权值为 w 的边，它能被计入 MST，当且仅当所有 $<w$ 的边都被计入 MST 后加入它不会造出一个环来。代码。

Kruskal 重构树

合并两个点集时，我们新建一个节点，权值为边权，得到的二叉树是基于边权的 Kruskal 重构树。

按照点权排序，遍历每个节点 u 和其能到达的节点 v，若 v 已经遍历，那么 $w_u\geq w_v$，将 v 的父亲设置为 u（如果不在一个集合内），所形成的多叉树是基于边权的 Kruskal 重构树。

重构树满足以下性质：

- 父亲节点的点权大于等于儿子的点权。
- 原图两点路径的瓶颈路等于树上的瓶颈路，即 LCA 处的权值。

Boruvka 算法

对于一个点 i，其最小权值的临边必定在 MST 上。那么迭代 $\log n$ 次，每次扫描每条边，然后合并连通块。

> CF888G.
>
> (i,j) 的边权是 $a_i \oplus a_j$，求 MST。

考虑一条一条边地合并，在 Trie 树上启发式合并，去找合并两个连通块地最小边权即可。

可以一开始将点权排序，这样连续的下标在 Trie 上是连续的。

* 最小生成树计数

需要使用 Matrix-Tree 定理，但是我是废物，不会大家都会的。

以后我会补的，我错了。

最小 mex 生成树

> 模板。
>
> 最小 mex 生成树。

直接线段树分治将答案分出来即可。

最小度限制生成树

> 模板。
>
> 最小生成树，但是要求 s 号点恰好连接了 k 条边。

恰好连接，而且看起来就是凸的，因此直接 wqs 二分即可。

我们能做得更好！我们可以求出所有 $f_x(i)-f_x(i-1)$ 并从小到大排序，取出前若干个即可得到答案。

实现上只需要先求出任意一棵生成树，然后贪心调整即可。

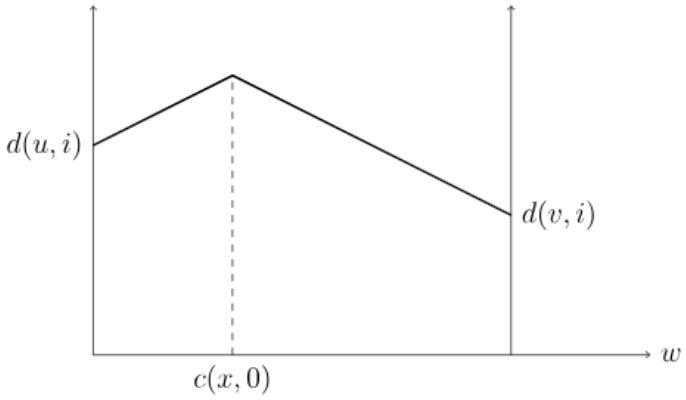
最小直径生成树

> https://www.luogu.com/article/26o6hkmm.

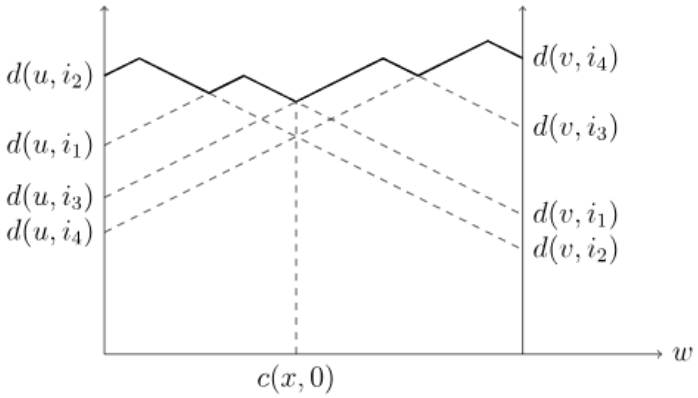
我们先介绍图的绝对中心。无向图的绝对中心位于图的边上或者节点上，满足该中心到所有点的最短距离的最大值最小。此时对应的最短距离最大值叫做直径。

那么使用 Floyd 算法，对于在边 (u,v,w) 上的情况，到点 i 的距离为 $\min\{d_{u,i}+x,d_{v,i}+w-x\}$：

随着图的绝对中心 c 在边上的改变会生成一个距离与 c 位置的函数图像。显然的，当前的 $d(c,i)$ 的函数图像是一个两条斜率相同的线段构成的折线段。



对于图上的任意一结点，图的绝对中心到最远距离结点的函数就写作 $f=\max\{d(c,i)\},i\in[1,n]$，其函数图像如下。



并且这些折线交点中的最低点，横坐标就是图的绝对中心的位置。

因此我们对于一条边 (u,v,w)，按照纵坐标枚举 i，然后如果最上面那个东西会产生交点的话，就更新最上面那个东西就可以了，此时用 $\dfrac{d_{u,i}+x+d_{v,p}+(w-x)}{2}$ 更新答案，注意计算两个端点。

对于最小直径生成树，求出图的绝对中心之后，由于图的绝对中心到最远点的距离最小，因此其所对应的半径就是最小直径生成树的半径，因此求出最短路树即可（注意初始距离），代码，注意边界点的特判。

k 小生成树

模板，仿照 k 短路的思路，我们来完成这个问题。

首先求出最小生成树。

对最小生成树求出权值增加量最小的非树边，则有两种选择：强制选择这条边，和强制不选择这条边。对于前者，给出了一棵新的生成树；对于后者，没有给出新的生成树，但边的状态改变了。

如果强制不选择这条边，那么接下来就是决策权值增加量次小的边是否强制选择，依次类推。于是最小生成树给出了若干棵新的生成树，其中第 i 棵生成树钦定了权值增加量前 i 小的非树边不选择，且第 i 小的非树边强制选择。将所有生成树加入优先队列，不断从优先队列取出权值最小的生成树 k 次做上述扩展即可。

注意已经钦定状态的边不可被改变。一条边有四种状态：树边且可被替换，非树边且可加入生成树，树边且不可被替换，非树边且不可加入生成树。在枚举所有非树边时需要跳过所有不可加入生成树的非树边，求一条非树边能替换掉哪条树边时不能替换不可被替换的树边。

如果将每棵生成树抽象为一个点，用一棵有根树描述整个扩展过程（根节点是最小生成树），那么每个非叶子结点有两个儿子，一个是实点，从当前点向下走到实点表示用一条非树边替换了一条树边得到新的生成树，替换出来的那条边变成了非树边且可加入生成树，替换进去的变成了树边不可替换；另一个则是虚点，从当前点向下走到虚点表示当前生成树的一条非树边从可加入变成了不可加入。

这种方式按照顺序枚举了所有的生成树（每条边都枚举了选还是不选），而且按照顺序扩展了次小生成树，因此这么做是对的。

为了避免直接扩展了 $O(m)$ 个虚儿子，应该在取出实点之后再加入兄弟虚点。时间复杂度 $O(km\log m+k\log k)$，运气好是能过的，代码。

官方题解没有采用倍增而采用了并查集维护次小生成树来做到 $O(m\log m+mk\alpha(n)+k\log k)$，我没看懂。

拟阵与生成树

可以使用带权拟阵交解决那种有满足一个限制条件然后权值最优的问题，至少对于笔者来说过于抽象了，强大的选手可以参考 IOI2018 集训队论文的《浅谈拟阵的一些拓展及其应用》学习。

但不是什么都需要拟阵！

CF1556H。给生成树定义估价函数 $f(T)=\sum_{i=1}^{K}\max\{0,D_i-d_i\}$，其中 D_i 代表实际度数。然后先求出最小生成树，随机调整即可。

其它生成树问题

最小欧几里得生成树需要使用特定的科技。

最小差值生成树，维护 LCT 即可。

最小乘积生成树，去年讲过，笔者很菜所以不会。

连通性问题

对于一张无向图，如果能够从 u 走到 v，那么称 u 和 v 是**连通的**。一个极大连通子图被称为**连通分量**。

对于一张有向图，将边替换成无向边可以连通则是**弱连通的**，不需要替换则是**强连通的**。同样可以定义连通分量的概念。

大部分的连通性问题都离不开 Tarjan 算法，因此我们先来回顾一下它的工作原理。它对每个结点 u 维护了以下值：

- 时间戳 dfn_u；
- 追溯值 low_u 代表 u 的子树中能够回溯到的最早的已经在栈中的结点。

在一个 SCC 中只有一个点满足 $dfn_u = low_u$，该结点一定是在深度遍历的过程中，该连通分量中第一个被访问过的结点。

根据此我们可以求出无向图的点双边双，有向图的 SCC。

无向图的双连通性

如果将 E' 从 E 中删去，G 便不再连通，那么 E' 是 G 的一个**边割集**，大小为 1 的边割集称为**割边/桥**。割边一定是树边，判定条件是 $dfn_u < low_v$。

如果 G 不存在大小为 $k-1$ 的边割集，则称 G 是 $k-$**边连通的**，k 最大时，称 k 为 G 的边连通度。

对于点也可以做同样的定义。割点的判定条件是 $dfn_u \leq low_v$，除了 Tarjan 的根节点，根节点需要有两个点满足这个条件；而有多少个点满足这个条件，就是割掉这个点之后图会分成多少个连通块。

- 对于边双内的任意两点 u, v，存在经过 u, v 的回路。
- 对于 ≥ 3 的点双中的两点 x, y（可以相等）与一条边 e，存在 $x \to e \to y$ 的简单路径；这可以说明存在经过 x, y 必定存在简单环，从 x 可以绕回自己。

广义圆方树

广义圆方树可以高效处理无向图双连通的相关问题。它是 v-DCC 缩点之后的产物。我们建出 v-DCC 的“代表方点”并向 v-DCC 内部所有点连边，这样会形成一个菊花图。

广义圆方树不能对应一棵唯一的仙人掌，因为非环边也会建一个方点。

- 圆点 x 的度数为它所在的 v-DCC 个数；
- 只有圆点和方点间有边；
- 如果方点的度数为 2，那么它连接的这两个点是原图的割边。值得注意的是，**广义圆方树判断割边无法处理重边的情况**，使用此性质判断割边时需要看一下 (x,y) 边是否为重边；
- 圆点 x 是叶子当且仅当它在原图上不是割点（否则连接了至少两个 v-DCC）；
- 广义圆方树上删掉圆点 x 后剩余节点的联通性与原图上删除 x 相等；
- x,y 简单路径上的所有圆点就是原图中 x,y 之间的所必经点。

> 「SWTR-8」地地铁铁
>
> 无向连通图，边有 01 权，无序点对 (x,y)（$x\neq y$）是好的当且仅当 x,y 之间存在同时出现 01 边权的简单路径。
>
> 给出一个 $O(n+m)$ 做法。

首先建出圆方树，如果 x,y 间有某个有 1 边的点双，那么它们一定能通过这条路径。

那么对于原图上一条 0 边 (u,v)，它恰属于一个点双 S 的方点打标记，统计圆方树上不经过标记点的圆点间路径数即可。这样能计数只经过 0 边的无序对。

看到这里感觉可以容斥了，也就是说要计数只有 0/1 路径，同时有 0 路径和 1 路径。

后者怎么做？可以发现这样的 (x,y) 点对一定在一个点双内。然后在这个点双内恰好满足只有这两个点即被 1 边连，又被 2 边连。

详细证明可以参考官方题解。

广义圆方树的好题有很多（往往和计数结合）：

- [省选联考 2023] 城市建造；
- [ZJOI2022] 简单题。

[ZJOI2022] 简单题

如果图是仙人掌，那么就根题目性质没什么关系了。因此每个点双独立，分析点双的性质。

分析可以得到，最多只有两个环相交，最终结论是一定存在两个点 S,T，使得这个点双可以由这两个点之间的若干条不相交的链组成，将其称为杏仁，或者有丝分裂期组成纺锤体的丝状结构（纺锤丝）。

那么对点 x 和 fa_{fa_x} 之间的简单路径权值和和方案数可以直接计算出来（树上前缀积预处理即可），这样可以直接跳到 LCA 然后去讨论。

网络流问题

一个**网络**是一张有向图 $G=(V,E)$，对于每条有向边 $(u,v)\in E$ 存在**容量限制** $c(u,v)$，当 $(u,v)\notin E$ 时，$c(u,v)=0$。网络的可行流分为有源汇（指定了两个节点 s,t，代表图的**源点**和**汇点**）和无源汇，但是都存在一个定义域为节点二元组的流函数 $f(x,y)$，$f(x,y)$ 代表边 $x\to y$ 的流量。

- 对于**有源汇**，有 $\sum f(S,i)=\sum f(i,T)$，此时这个相等的和成为当前流 f 的流量。
- 定义流 f 在网络 G 上的**残量网络** $G_f=(V,E_f)$ 为容量函数 $c_f=c-f$。根据容量限制，$c_f(x,y)\geq 0$，当 $c_f(x,y)=0$ 时，则视为 $x\to y$ 在残量网络上不存在。也就是说，在残量网络中我们要删掉满流边。
- 定义**增广路** P 是残量网络 G_f 上源点到汇点的一条路径，而无源汇则没有增广路。
- 将点集分为两个互补相交的 A,B，且满足 $S\in A,T\in B$，这种划分方式称为**割**，割的容量为 $\sum_{u\in A}\sum_{v\in B}c(u,v)$，流量为 $\sum_{u\in A}\sum_{v\in B}f(u,v)$。如果 $u\in A,v\in B$，那么 (u,v) 是**割边**，可以看出，割边一般不止一条。
- 割中割边容量和最小的划分方式称为**最小割**，而且**最大流等于最小割**。

最大流问题

解决最大流问题的思想是：**不断寻找增广路** 和 **能流满就流满**。在给一条边增加流量时，我们需要给其反向边来增加容量来支持反悔。这样的操作称为一次**增广**。

EK 算法的思想是使用 BFS 寻找**长度最短**的增广路，然后计算流量。为此我们记录流向每个点的边的编号，然后从汇点反推到源点，时间复杂度为 $O(nm^2)$。

Dinic 的本质思想是多路增广和当前弧优化（跳过流满的边）。通过 BFS 将图分层，向下一层节点开始进行多路增广，并记录当且流满到了哪条边。时间复杂度 $O(n^2m)$，在二分图上是 $O(m\sqrt{n})$ 的。

将 BFS 改成 SPFA 即可解决无负环的费用流问题，此时 Dinic 时间复杂度是 $O(nmf)$ 的，其中 f 是最大流流量。

上下界网络流

To Do

常见模型

例题

[TTPC2022] Colorful Graph

> Portal.
>
> 有向图，要求给点染色，使得同色点 i,j 存在一条 $i\to j$ 或 $j\to i$ 的道路（或者都有），要求颜色数最小，给出方案。
>
> $n,m\le 7\times 10^3$，8s。64 MiB。

首先 SCC 缩点，一个 SCC 内可以染同一个颜色，那么只需要处理 DAG。

求的是 DAG 最小可交路径覆盖。也就是一个点内部的流量是不限的。

图论杂项

弦图

参考 https://oi.wiki/graph/chord，对于笔者来说过于抽象了。

仙人掌与圆方树

感觉这套技术还是太闹鬼了。

模拟费用流

参考 IOI2022 集训队论文《模拟费用流及其应用》。

实际上该部分需要大致了解，但限于时间，笔者并未准备完成这一部分的内容，请各位谅解。

针对建图的分析

优化建图

数据结构优化建图：如果图中边数太多，则可以尝试使用数据结构优化建图的方式减少边数。常见的数据结构优化建图的方式有前后缀优化建图、线段树优化建图、主席树优化建图和堆优化存边等。

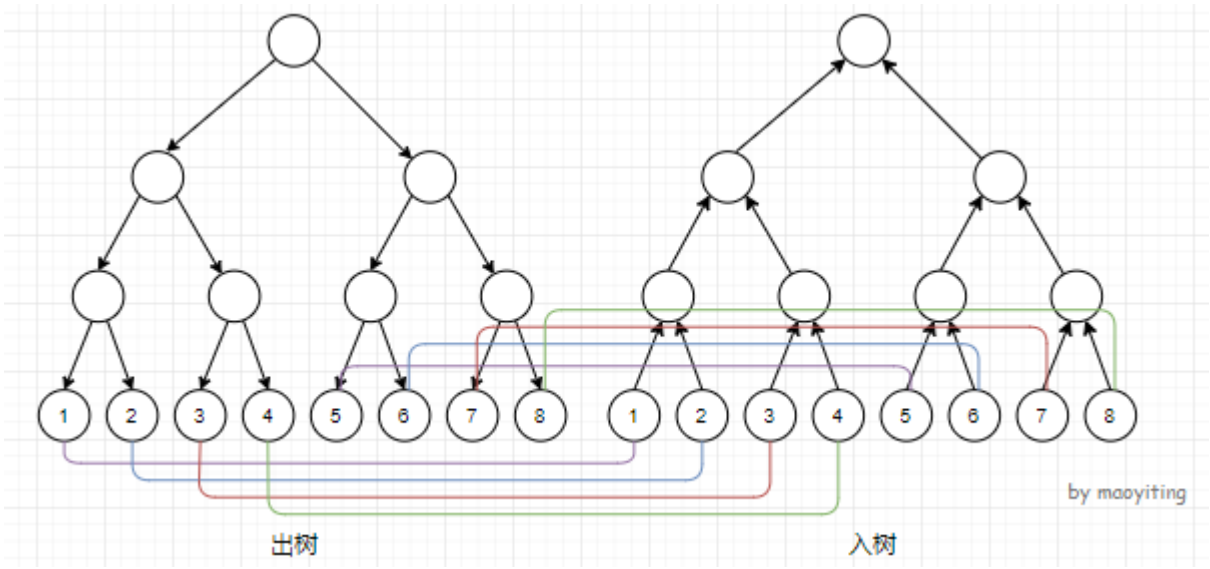
> [PA2012] Tax.
>
> 给出一个 n 个点 m 条边的无向图，经过一个点的代价是进入和离开这个点的两条边的边权的较大值，求从起点 1 到点 n 的最小代价。起点的代价是离开起点的边的边权，终点的代价是进入终点的边的边权。$1 \le n \le 10^5$，$1 \le m \le 2 \times 10^5$，$1 \le c \le 10^6$。

边走到边有代价，那干脆把边转成点，一条有向边对应着一个点，两条反向边的边权为原边权（切边的贡献），然后通过枚举中转点的入边出边来建边就可以直接算出贡献，这样的边数是这样的边数是 $O(m^2)$ 的。

常用的优化思路是，**将边转化为等效的内容**。实际上我们可以对于一个点的出边按照边权从小到大排序后差分依次连接，反着连则是边权为 0。这样在从入边切到比入边边权小的出边时就没有额外的贡献，而切到比它大的东西时就会在走这条差分链时产生贡献，刚好可以满足条件。

> CF786B.
>
> 区间连边，最短路。

建立两棵线段树，大概像这样：



于是直接在线段树上连边即可。

针对图的性质的分析

随机化方法

当某些基本信息的数量超出预期，或者问题可以近似看成某些 NPH 问题时，我们可以考虑随机化来解决。

[THUSCH2017] 巧克力。

中位数可以直接一个二分处理掉。

一看，这不是斯坦纳树吗！但是发现做不了，颜色数爆炸了。

随机赋值颜色，没了。

这道题本来在杂题里，但是太典了，所以给放到这里来了，以下是原题解。

如果颜色数比较少的话直接用斯坦纳树做，但是颜色数很多，钦定的可能也很多。

k 很小，因此考虑将所有颜色随机映射到 $[0,k)$，然后求最小斯坦纳树即可求出最小的巧克力个数 w。这 k 个点被分配到不同的颜色时答案合法，正确概率是 $k!/k^k$。随机化做 200 次即可，这样就求出了块数。

然后二分出中位数，将小于等于二分值的权值都设为 $inf-1$，大于的都设为 $inf+1$，然后最小斯坦纳树要 $\leq w\times inf$（inf 设置为一个不会影响斯坦纳树选择的巧克力数的一个数即可）。

杂题选讲

CNOI 的有些题目需要更多的自行思考，直接在课上讲了这些题会严重破坏它们的思考体验，因此今天并没有选择讲解这些题目。这里将一些比较有意思的思维题（而且无需深入分析挖掘性质）的题拉出来给大家开心一下。难度不会很大。

[ARC098D] Donation

> Portal.
>
> 无向连通图，每个点有两个权值 A_i,B_i。
>
> 最开始时你拥有一定数量的钱，并且可以选择这张图上的任意一个点作为起始点，之后你从这个点开始沿着给定的边遍历这张图。每当你到达一个点 v 时，你必须拥有至少 A_v 元。而当你到达了这个点后，你可以选择花 B_v 元打卡（当然也可以选择不打），当然，你需要保证在打卡之后自己剩余的钱 ≥ 0。
>
> 你需要对所有的 n 个点都打卡一次，求你一开始至少需要携带多少钱。
>
> $n\leq 10^5$。

首先，在某点打卡后必不会再次访问某点。

因为若两次访问某点，第一次不打卡，第二次打卡要比第一次就打卡优，因为中间的一段路程余钱更多。

因此从一个点 i 出来时，剩余的钱不会少于 $c_i=\max\{a_i-b_i,0\}$。所以按照 c_i 从小到大排序，建立点权重构树。树形 DP 的策略是先走度数减一个子树，然后打卡自己，再走最后一个子树。这样树形 DP 可以覆盖所有的策略。

[HNOI2019] 校园旅行

> Portal.
>
> $n \leq 5000, m \leq 5 \times 10^5$，点有 01 权值，对于每个点对求出其是否存在回文路径。

考虑暴力，$f_{x,y}$ 是否存在 $x \to y$ 的回文路径，直接记忆化搜索 $O(n^2 + m^2)$。

问题是我们的边数太多了！注意到路径长度其是不太要紧，因为可以来回走刷分。首先考虑一个事情，如果我只能走同色点，那么我们好像不能改变我们当前刷的路径长度的奇偶性——除非有奇环，即不是二分图。

那么对于二分图的同色连通块，可以只保留一棵生成树（因为子图也是二分图），其它同色连通块可以表示为一棵生成树，然后有一个自环。

而异色点是自然二分图，直接保留生成树即可。

[CF325E] The Red Button

> Portal.
>
> 我们找到了最后的按钮，你只有最后一次机会改变命定的终焉。
>
> 有 $n(n \leq 10^5)$ 个节点，编号 $0 \sim n-1$，需要按照一定顺序拆毁它们，在拆毁 i 节点后，只能拆毁 $2i \bmod n$ 或 $(2i+1) \bmod n$ 节点，最后需要再拆毁一次 0 节点，剩下的节点都要被拆毁恰好一次。
>
> 找到那最后一块拼图，或者报告无解。

这是个什么？不知道。先打表找规律，然后发现 n 为奇数时无解。

然后呢？以下是 n 为偶数时暴力跑出的答案：

```
0 1 0
0 1 3 2 0
0 1 2 5 4 3 0
0 1 2 5 3 7 6 4 0
0 1 2 4 9 8 6 3 7 5 0
```

能看出什么吗？好像不能，那再从图上出发看看能不能发现点什么。

每个点的出度均为 2，要是 1 我直接欧拉路！诶可不可以通过特殊限制搞成 1 呢？还真能。

我们需要一个更好的代数形式来刻画每个点的出边。我们有：

$$i \times 2 \equiv (i + \frac{n}{2}) \times 2 \pmod{n}$$
$$i \times 2 + 1 \equiv (i + \frac{n}{2}) \times 2 + 1 \pmod{n}$$

也就是说 i 和 $i + \frac{n}{2}$ 所连的边应该是同一个东西。那这好办了，想要有答案必须一人分一个，那就随便。

然后答案成了什么状物？若干个环！我们需要合并它们。

如果图上有多个环，那么必定有一对等效点不在同一个环中，交换它们分的边即可合并环。

我们已经通过暴力验证了 n 为偶数时一定有解，所以这样一定可以合并所有的环（否则没得可走了）。

[POI2014] RAJ-Rally

> Portal.
>
> 一个 5×10^5 的无权 DAG，选择一个点使得删掉这个点后的最长路最短。

考虑按照拓扑序 $1 \sim n$ 枚举每个点，开始时所有点在 B 集合内，内部有 n 个距离，然后把拓扑序小的点加到 A 集合里，每次维护可以穿过 A, B 集合的距离即可。

[CTT2012] 最小生成树

> Portal.
>
> 无向连通图并额外给定一条边 (u, v, L)，问需要删掉原图多少条边才能使得这条边既能出现在最小生成树又能出现在最大生成树。
>
> $n \leq 2 \times 10^4, m \leq 2 \times 10^5$。

最小生成树是什么？将 $< L$ 的边都加入，(u, v) 不连通。

两个最小割加起来即可。

[CF1835F] Good Graph

Portal.

无解直接 Hall 定理判断即可。

对于有解，设 S_i 代表 i 的最小紧密集合，其过程是不断地跑交替路，然后跑到一定程度它会闭合。也就是说，我们只需要在 $P(i) \neq j$ 时，也就是 $mch_i \neq j$ 时，连一条 $i \to mch_j$ 的边。

也就是说我们要找一张最小图使得和原图传递闭包相同，先 `bitset` 求出，SCC 内部连环，外面依次连接即可。