最小生成树

李淳风

长郡中学

2024 年 9 月 27 日

前言

给定一张边带权的无向图 $G=(V,E), n=|V|, m=|E|$。由 V 中全部 n 个顶点和 E 中 $n-1$ 条边构成的无向连通子图被称为 G 的一棵生成树。边的权值之和最小的生成树被称为无向图 G 的最小生成树。
定理：任意一棵最小生成树一定包含无向图中权值最小的边。
假设无向图中存在一棵最小生成树不包含权值最小的边。设 $e=(x,y,z)$ 是无向图中权值最小的边。把 e 添加到树中，e 会和树上从 x 到 y 的路径一起构成一个环，并且环上其它边的权值都比 e 大。因此，用 e 替代环上任意一条边，会形成一棵权值和更小的生成树，与假设矛盾。
由此我们可以得到一个推论。给定一张无向图 $G=(V,E), n=|V|, m=|E|$。从 $|E|$ 中选出 $k<n-1$ 条边构成 G 的一个生成森林。若再从剩余的 $m-k$ 条边中选 $n-1-k$ 条添加到生成森林中，使其成为 G 的生成树，并且选出的边权之和最小，则该生成树一定包含这 $m-k$ 条边中连接生成森林的两个不连通节点的权值最小的边。

Kruskal 算法

Kruskal 算法就是基于上述推论的。Kruskal 算法总是维护无向图的最小森林。最初，可以认为森林由零条边构成，每个节点各自构成一棵仅包含一个节点的树。
在任意时刻，Kruskal 算法从剩余的边中选出一条权值最小的，并且这条边的两个端点属于生成森林中两颗不同的树（不连通），把该边加入生成森林。图中节点的连通情况可以用并查集来维护。
Kruskal 算法的具体流程如下：

- 建立并查集，每个点各自构成一个集合。
- 把所有边按照权值从小到大排序，依次扫描每条边 (x, y, z)。
- 若 (x, y) 属于同一集合（已经连通），则忽略这条边，继续扫描下一条。
- 否则，合并 (x, y) 所在的集合，并把 z 累加到答案中。
- 所有边扫描完成后，之前累加过的边就构成最小生成树。

时间复杂度为 $O(m \log m)$。

Prim 算法

Prim 算法同样基于上述推论，但思路略有改变。Prim 总是维护最小生成树的一部分。最初，Prim 算法仅确定 1 号节点属于最小生成树。

在任意时刻，设已经确定属于最小生成树的节点集合为 T，剩余节点集合为 S。Prim 算法找到 $\min\limits_{x\in S, y\in T}\{z\}$，即两个端点分别属于集合 S, T 的权值最小的边，然后把点 x 从集合 S 中删除，加入到集合 T，并把 z 累加到答案中。

具体来说，可以维护数组 d：若 $x \in S$，则 $d[x]$ 表示节点 x 与集合 T 中的节点之间权值最小的边的权值。若 $x \in T$，则 x 就等于 x 被加入 T 时选出的最小边的权值。

类似 Dijkstra 算法，用一个数组标记节点是否属于 T。每次从未标记的节点中选出 d 值最小的，把它标记并新加入 T，同时扫描所有出边，更新另一个端点的 d 值。最后，最小生成树的权值总和就是 $\sum_{x=2}^{n} d[x]$。

Prim 算法的时间复杂度为 $O(n^2)$，可以用二叉堆优化到 $O(m \log n)$，但这样不如直接使用 Kruskal 算法方便。因此 Prim 主要用于稠密图，尤其是完全图的最小生成树的求解。

例题

走廊泼水节

给定一棵 N 个节点的树，要求增加若干条边，把这棵树扩充为完全图，并满足图的唯一最小生成树仍然是这棵树。求增加的边的权值总和最小是多少。$N \leq 6000$，所有边权均为非负整数。

走廊泼水节

N 个点的树有 $N-1$ 条边，把这些边按照权值从小到大排序，依次扫描每条边，执行一个类似于 Kruskal 算法的过程。
设当前扫描到边 (x,y,z)，x 所在的并查集为 S_x，y 所在的并查集为 S_y，此时应该合并 S_x 与 S_y。合并后的集合 $S_x \cup S_y$ 构成一棵树的结构。
$\forall u \in S_x, v \in S_y$，若 $(u,v) \neq (x,y)$，则在最终的完全图中，我们肯定需要在 (u,v) 之间增加一条边。于是，无向边 (u,v)、S_x 中从 u 到 x 的路径、无向边 (x,y)、S_y 中从 y 到 v 的路径构成了一个环。要想让 (u,v) 在最终的最小生成树上保留，我们至少需要给 (u,v) 的边权定为 $z+1$。而 S_x 与 S_y 之间一共会增加 $|S_x| * |S_y| - 1$ 条边，因此每次我们把 $(z+1) * (|S_x| * |S_y| - 1)$ 累加到答案当中即可。

例题

Picnic Planning

给定一张 N 个点 M 条边的无向图，求出无向图的一棵最小生成树，满足 1 号节点的度数不超过给定的整数 S。$N \leq 30$。

例题

Picnic Planning

给定一张 N 个点 M 条边的无向图，求出无向图的一棵最小生成树，满足 1 号节点的度数不超过给定的整数 S。$N \le 30$。

首先，去掉 1 号节点之后，无向图可能会分为若干个连通块。先用 DFS 或并查集划分出图中的每个连通块。设连通块有 T 个，若 $T > S$，则本题无解。
对于每个连通块，在这个连通块内部求出它的最小生成树，然后从连通块中选出一个节点 p 与 1 号节点相连，其中无向边 $(1, p)$ 的权值尽量小。
此时，我们已经得到了原无向图的一棵生成树，1 号节点的度数为 T。我们还可以尝试改动 $S - T$ 条边，让答案更优。
考虑无向图中从节点 1 出发的每条边 $(1, x)$，边权为 z。如果 $(1, x)$ 还不在当前的生成树中，那么继续找到当前生成树中从 x 到 1 路径上权值最大的边 (u, v)，权值为 w。求出使得 $w - z$ 最大的点 x_0，若 x_0 对应的 $w_0 - z_0 > 0$，则从树中删掉边 (u_0, v_0)，加入边 $(1, x_0)$，答案就会变小 $w_0 - z_0$。
重复上一步 $S - T$ 次，或者直到 $w_0 - z_0 \le 0$，就得到了题目所求的最小生成树。

例题

最优比率生成树

给定一张 N 个点、M 条边的无向图 $(1 \leq N, m \leq 10000)$，图中每条边 e 都有一个收益 C_e 和一个成本 R_e。求该图的一棵生成树 T，使树中各边的收益之和除以成本之和，即 $\frac{\sum_{e \in T} C_e}{\sum_{e \in T} R_e}$ 最大。

例题

最优比率生成树

给定一张 N 个点、M 条边的无向图 $(1 \leq N, m \leq 10000)$，图中每条边 e 都有一个收益 C_e 和一个成本 R_e。求该图的一棵生成树 T，使树中各边的收益之和除以成本之和，即 $\frac{\sum_{e \in T} C_e}{\sum_{e \in T} R_e}$ 最大。

这实际上是一道 0/1 分数规划问题。我们二分一个答案 mid，新建一张结构不变的完全图，每条边的权值变为 $C_e - mid * R_e$。若该图上的最大生成树边权值和非负，则说明存在答案大于等于 mid 的生成树，令 $l = mid$，否则令 $r = mid$。达到题目要求的精度时二分结束。

例题

黑暗城堡

黑暗城堡有 $N(1 \le N \le 1000)$ 个房间，M 条可以制造的双向通道，每条可以制造的通道都有一个非负的长度。
Lord lsp 想把城堡修建成树形的，同时他还会使得城堡满足下面的条件：设 $D[i]$ 为如果所有路径都被修建，第 i 号房间与第 1 号房间的最短路径长度；而 $S[i]$ 为在实际修建的树形城堡中第 i 号房间与第 1 号房间的最短路径长度；要求对于所有的整数 $i(1 \le i \le N)$，$S[i] = D[i]$ 成立。
请找出有多少种修建的方案可以满足 Lord lsp 的要求。答案对 $2^{31} - 1$ 取模。

黑暗城堡

首先，我们可以用 DIjkstra 算法求出 1 号房间到每个房间的单源最短路，把树形城堡看作以 1 为根的有根树。根据题目要求，若 x 是 y 的父节点，x, y 之间的通道长度为 z，则应该有：$dist[y] = dist[x] + z$。
事实上，我们把满足题目要求的树结构，即对任意一对父子节点 x, y 都有上式成立的树结构，称为图的一棵最短路径生成树。
我们可以把所有节点按照 $dist$ 从小到大排序，依次考虑每个节点 p 加入树形城堡有多少种方法。与 $Prim$ 算法类似，我们维护 "最短路径生成树" 的一部分，记为 T。统计有多少个节点 x 满足 $x \in T$ 且 $dis[p] = dis[x] + edge(x, p)$，其中 $edge$ 表示边的长度。让 p 与任意一个 x 相连都符合要求。
根据乘法原理，我们把每一个点可选择的方案数乘起来，就得到了整道题目的答案。