《通道》试题讲评

清华大学交叉信息研究院 陈俊锟 吕欣

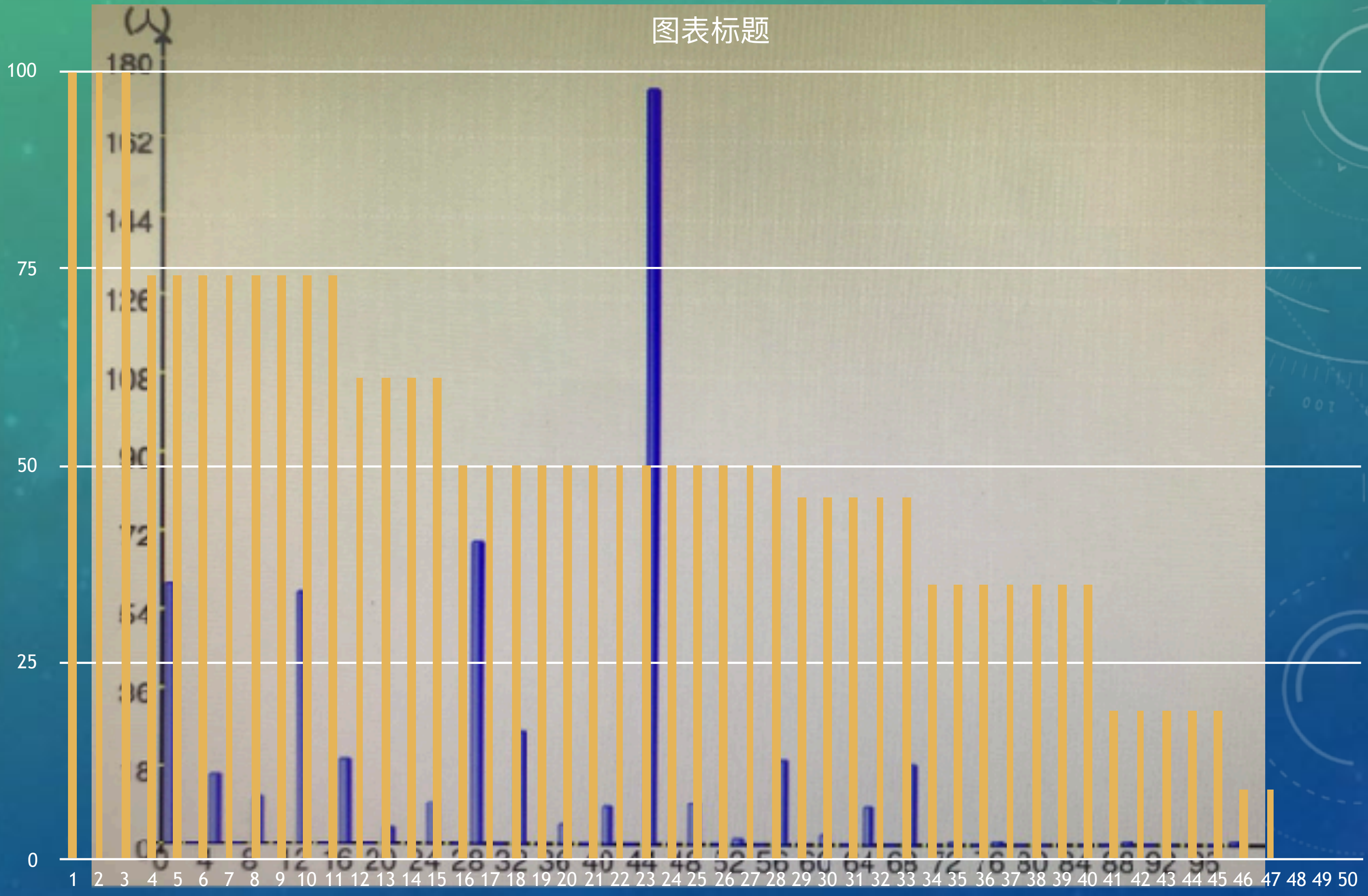
题意简述

- 给出三棵树 T_1, T_2, T_3，每条边有非负边权
- 求一对点 a, b，使得 $d_1(a,b) + d_2(a,b) + d_3(a,b)$ 最大
- 其中 $d_i(a,b)$ 表示 a, b 在 T_1 中的距离

- 既然题意这么简单，这题肯定也非常简单，让我们看看得分情况

得分情况

- 集训队
  - AC * 3
  - 平均分 48.45
- 非集训队
  - AC * 1



吐槽

BF：暴力

- 枚举 a, b
- 用求 LCA 来求树上距离
- LCA 可以 $O(\log n)$ 地求，不必 $O(1)$

- $O(n^2 \log n)$ 28pts

HACK1

- 猜测：答案的 (a, b) 一定在某一个 T_i 中是比较长的一条链
- 取出每个 T_i 的前 $k = k(n)$ 长链，更新答案
-

- $O(k(n)\log^2 n)$ 或 $O(k(n)\log n)$ 24~76pts

HACK2

- 重复很多次以下操作
  - 从一个点 a 出发，找另一个点 b 使得答案最大
  - 从 b 出发，找一个点 c 使得答案最大，更新答案
- 通过卡时运行尽量多次

- By LCA
- $O(n)$ ~~100~~97pts

HACK3 ~ HACK∞

- 欢迎介绍~

数据大类

- S1：链（4pts）
- S2：树（20pts）
- S3：树 + 链（12pts）
- S4：树 + 树（8pts）
- S5：树 + 链 + 链（12pts）
- S6：树 + 树 + 链（8pts）
- S7：树 + 树 + 树（36pts）

S1：链

- 只有一条链，必然取两个端点
- 答案就是所有边权加起来的和
-

- $O(n)$ 4pts

S2：树

- 等价于求最长链
- 由于边权非负，可以用贪心
  - 随便选一个点 DFS 整棵树算距离，找出最远点 a
  - 从 a 出发 DFS 整棵树算距离，找出最远点 b
- 也可以 DP
  - 记录每个点子树中的最深和次深节点

- $O(n)$ 24pts = S1 + S2

S3：树 + 链

- 在 LCA 处考虑所有 T_1 中 LCA 为它的点对
- 如果 $LCA_1(a,b)=p$，那么我们就是要最大化 $(h_a+h_b+|l_a-l_b|)-2h_p$
  - h_i 是 i 在 T_1 中的深度，l_i 是 i 在 T_2 中距离端点（节点 1）的距离
  - h_p 和 a,b 无关，只需要考虑另外几项
  - 记录 h_i+l_i 和 h_i-l_i 的最大、次大值，选择两个不同的点加起来
    - $|x|=\max(x,-x)$，如果取的符号不对肯定不优，因此只需如此简单考虑

- $O(n)$ 36pts = S1 + S2 + S3

S4：树 + 树

- 同样在 LCA 处考虑所有 T_1 中 LCA 为它的点对
- 现在问题转化为最大化 $\left(h_a + h_b + d_2(a, b)\right) - 2h_p$
- 怎么处理这个式子呢？

S4：树 + 树

- 设法将 h_i 的贡献“附着”在 T_2 上
- 对于 T_2 内的每一个点 i，新建 i' 和 i 边权为 h_i 的边，设新的树为 T_2'
  - 立刻可知 $d_2'(a,b) = [a=b](h_a+h_b)+d_2(a,b)$
  - 我们还有 $a \neq b$，所以最大化目标就是 $d_2'(a,b)$
- 题目转化为：给两个集合 A,B，求 $a \in A, b \in B$ 最大化 $d_2'(a,b)$
  - A,B 对应 T_1 中 p 的不同子节点的子树
  - 要支持集合的合并

S4：树 + 树

- 启发式合并显然是可以的，但是太复杂也太慢
- 对于边权非负的图，我们有一个性质
  - 跨越集合 A, B 的最长链的端点一定是 A 中最长链端点和 B 中最长链端点
  - 由反证法立刻可知结论成立
- 只需要在并查集中记录最长链端点，合并时更新答案即可
  - 合并后集合的最长链 = max(跨越, 内部)
- $O(n\log n)$ 或 $O\big(n\alpha(n)\big)$（取决于 LCA 算法）44pts = S1 + ⋯ + S4

S5：树 + 链 + 链

- 仍然是 LCA 处考虑路径
- 在这里问题是转化为 “曼哈顿距离最远点”
  - 同样只需拆掉两个绝对值符号，记录 2×2 种和或差的最大值即可
- 注意这里没必要也不能显式地建出 T_2'，只需要计算 $d_2'(a,b)$ 即可
- $O(n)$ 48 = S1 + S2 + S3 + S5

S6：树 + 树 + 链

- 从这里开始，题目已经变得很复杂了
- 但链的情况最简单，从链入手考虑
- 考虑将链的限制消去，转化为前面的“树 + 树”的情况

S6：树 + 树 + 链

- 对链进行分治，每次取中点，统计所有过中点的路径
- 现在问题转化为
  - 在中点的左侧和右侧分别找一个点
  - 使得 $d_1(a,b) + d_2(a,b) + l_a + l_b$ 最大
  - l_i 为 i 到中点的距离
- 将 l_i 附着到 T_2 上构造 T_2'，然后在 T_1 和 T_2' 上执行树 + 树的算法
- 做完了……吗？
- 复杂度 $O(n^2)$ ……

S6：树 + 树 + 链

- 为什么复杂度炸了？
  - 每次要完整地遍历 T_1 和 T_2'
  - 每个点都会被作为中点
- 我们为什么要完整地遍历，而不是只遍历“所需要的点”？
- 我们当然可以这么做！

S6：树 + 树 + 链

- 对分治的区间中的所有节点建立 T_1 的**虚树**
  - 虚树上包括所有“关键点”两两的 LCA
  - 从而可以使用在 LCA 处考虑路径的算法
- 并查集的部分，可以只清空、维护关键点的父亲
  - 其他点的信息根本就不会被使用

- $O(n\log^2 n)$ 或 $O\big(n\log n\,\alpha(n)\big)$ 64pts = S1 + ⋯ + S6

S7：树 + 树 + 树

- 结合 S6 和 BF，我们已经拿到了 84 分
- 在考场上这是完全可行且推荐的策略
- 剩下的 16 分怎么拿呢?

S7：树 + 树 + 树

- 考虑将链的处理方式扩展到树上
- 能否像链（序列）上的分治那样“找中点”然后“考虑过中点”？
- 考虑点分治，分治后我们要在不同的重心的分支找到两个点
  - 维护“包含分支 i”和“包含全部减分支 i”的集合的最长路
  - 由于重心的分支可能很多，不能像链上暴力维护属于哪个分支了
  - 使用 map 或 hash 进行启发式合并
-

- $O(n\log^3 n)$ 或 $O(n\log^2 n)$ 84~100pts

S7：树 + 树 + 树

- 如何减少分支个数呢？修改树的形态！
  - 对每个点用类似“左儿子右兄弟”的方法进行转化
  - 转化后任意两个原来树上点的距离不变
  - 但每个点最多只有 3 个分支
- 这样就可以暴力记录，不再需要启发式合并了

- $O(n\log^2 n)$ 或 $O\big(n\log n\,\alpha(n)\big)$ 100pts

常数问题

- 善良的 zgg 强行要求标程 5 倍 TL、标程 4 倍 ML
- 卡常数？不存在的！

- 那为什么我还是 T 了？

总结

- 本题题意非常简洁，做法较为直观
- 考察了点分治、虚树和并查集等常见的简单知识点和少量思维能力
- 是一道中规中矩的 NOI 中等难度试题，属于 WC 中的简单题
- 希望能在寒冷的冬天给大家带来一丝温 ♂ 暖

EOF

- 感谢 CCF 给了我这个命题和交流的机会
- 感谢 access_globe 给我原型（树 + 树、树 + 链 + 链）并和我讨论
- 感谢张哥哥帮我验题

- 欢迎提问
- 祝大家新年快乐、学业有成！