CTT2023 Altar

学习使用 TikZ 的 ltst

2023 年 12 月 5 日

题意

简要题意

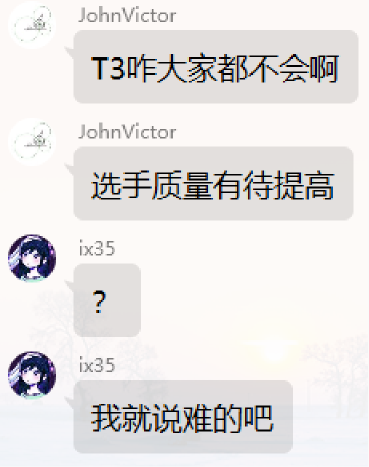
给定一个 n 边形的三角剖分，n 边形的点编号不一定按顺序。做以下三种操作将三角剖分的边删空。

- 询问两点之间的最短路，认为所有边长度为 1。满分要求使用不超过 $3.5n$ 次该操作。
- 删掉一条存在的边。
- 恢复一条之前被删过的边。你不能使用超过 $m = 2n - 3$ 次该操作。

$n \leq 1000$

吐槽


JohnVictor
noitamixorppa 10:38:02
jv 觉得 t3 要做多久。
@noitamixorppa 我做题 30min
JohnVictor
写代码不知道
哼!
JohnVictor
感觉熟练的选手 0.5h

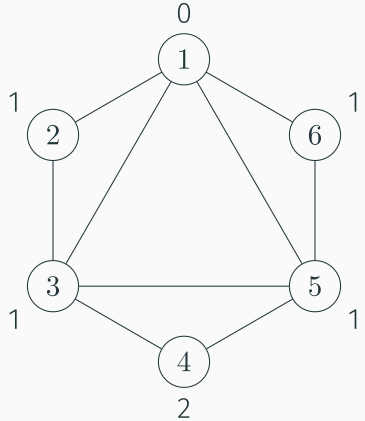

JohnVictor
T3咋大家都不会啊
JohnVictor
选手质量有待提高
ix35
?
ix35
我就说难的吧

正解

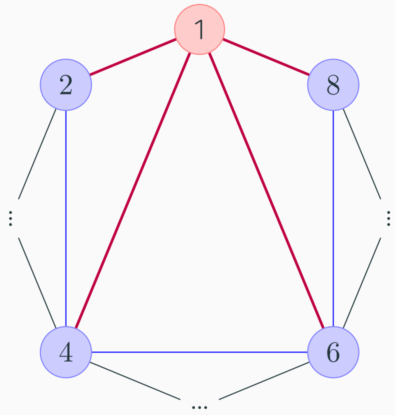
正解

花费 n 次询问，按照 $\text{dist}(1,x)$ 对所有点 x 分层，然后一层一层地考虑。

（为了方便演示，我们假设点按多边形顺序编号，但做法并不依赖这一点)



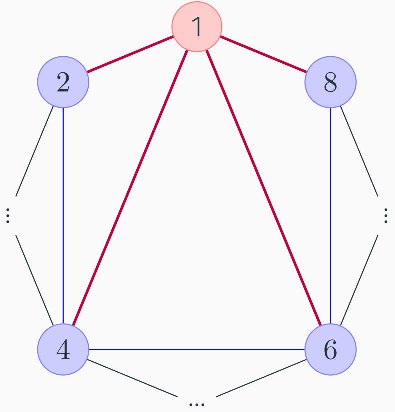
正解——加第二层



我们首先考虑加入第二层（蓝点）。此时我们需要考虑两种边：

- 第一层与第二层的边（紫色边）
- 第二层内部的边（蓝色边）

正解——加第二层

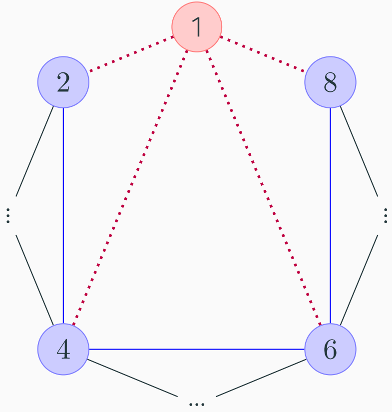


我们首先考虑加入第二层（蓝点)。此时我们需要考虑两种边：

- 第一层与第二层的边（紫色边)
- 第二层内部的边（蓝色边)

注意到紫色边是已知的，蓝色边构成一条链。

正解——加第二层



故我们把紫色边删掉，此时由于原图是三角剖分，任意两个蓝点的最短路都在蓝色链上。

正解——还原链

现在问题就变成了，给定一条链，通过问两点距离进行还原。

正解——还原链

现在问题就变成了，给定一条链，通过问两点距离进行还原。

这是比较简单的：

- 任选链上一点 x，按照 $\mathrm{dist}(x, y)$ 对所有 y 进行分层。此时每层最多只有两个点。
- 简单的分类讨论可以得到，相邻两层之间只需要通过至多一次询问就可以得到它们的连边情况。

正解——还原链

现在问题就变成了，给定一条链，通过问两点距离进行还原。

这是比较简单的：

- 任选链上一点 x，按照 $\mathsf{dist}(x, y)$ 对所有 y 进行分层。此时每层最多只有两个点。
- 简单的分类讨论可以得到，相邻两层之间只需要通过至多一次询问就可以得到它们的连边情况。

当链的点数为 m 时，该做法可以以 $\lceil 1.5m \rceil$ 次询问还原链。

注意到在子任务 1（有一个点度数为 $n-1$）中，可以在 $n + O(1)$ 次询问内得到度数最大的点，然后转化为还原链的情况，因此到该步可以获得 10 分。

正解——分治?

接下来一个很自然的想法是将删去红色边产生的若干个子三角剖分分治处理。

不过看起来这个东西在判断每个点属于哪个三角剖分上有些细节，而且似乎排除不掉 $\log$ 因子，常数也很大。

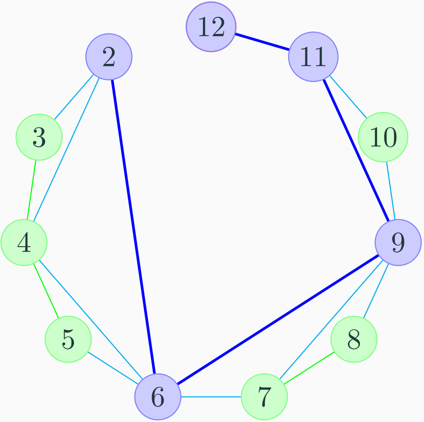
~~出题人摆了所以也不知道能拿多少分。~~

正解——继续加第三层

分治做法看起来前途不是很明朗，我们考虑直接继续做。考虑加入第三层的点，并识别出新增的边。

此时我们保留第二层的点之间的连边（即蓝色链），把 1 给扔掉。

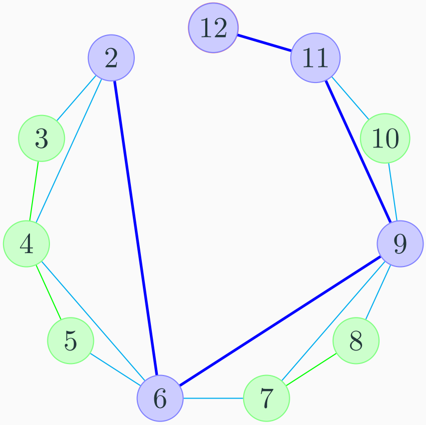
正解——理解新一层加入的边



我们先通过画图理解一下第三层（绿点）加入之后的新边长啥样。在原有蓝色边上，我们加入了

- 第二层和第三层之间的边（青色）
- 第三层之间的边（绿色）

正解——理解新一层加入的边

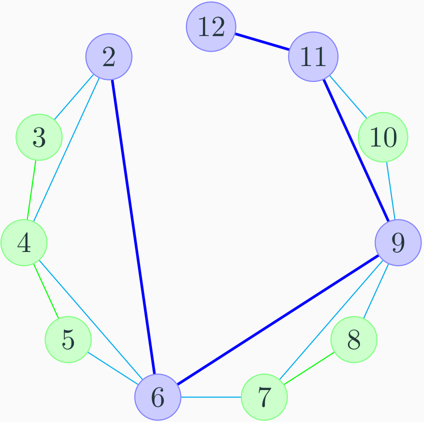


我们先通过画图理解一下第三层（绿点）加入之后的新边长啥样。在原有蓝色边上，我们加入了

- 第二层和第三层之间的边（青色）
- 第三层之间的边（绿色）

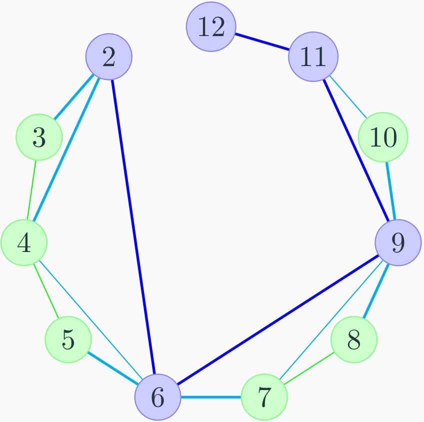
注意到以下事实：一条蓝色边会对应一些绿色点。这一串绿色点的绿色边构成一条链，且链的一段前缀连接蓝色边的一端，一段后缀连接另一端，前缀和后缀的交集大小恰好为 1。

正解——初步寻找蓝色边



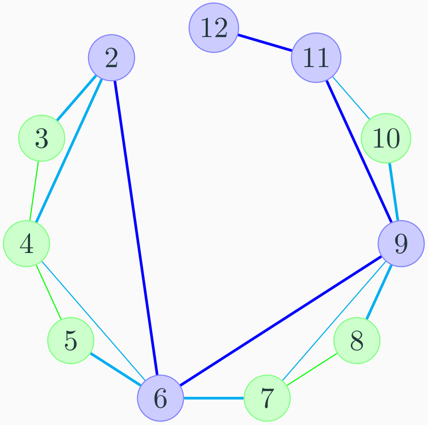
我们先考虑找（一些）青色边。
注意到任意一个绿色点到任意一个蓝色点的最短路都一定是一条青色边加蓝色链的一段。
于是我们只需要对每个绿色点询问到蓝色链的一端就可以找到与绿色点相邻的一条青色边。

正解——初步寻找蓝色边



这里我们选端点为 2，于是新加粗的青色边就是找到的边。
当然你也可以同时问两端得到所有青色边，但这样比较浪费。

正解——初步寻找蓝色边



这里我们选端点为 2，于是新加粗的青色边就是找到的边。

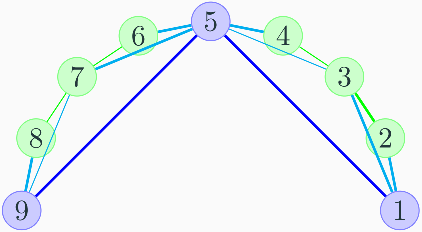
当然你也可以同时问两端得到所有青色边，但这样比较浪费。

接下来注意到，我们只需要把青色粗边构成的连通块排好就可以了。

按照蓝色链上的顺序，对每个青色粗边的连通块分别排顺序。

正解——排连通块

考虑一种可能的时局图，考虑 5 所在的连通块：

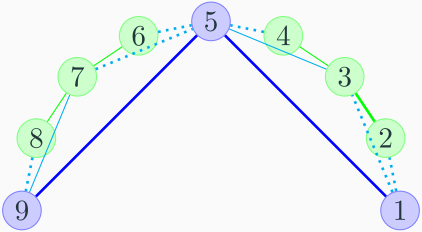


此时，其前驱 1 所在的连通块已经排好了，所以 $(3,5)$ 的青色边也是呼之欲出的。

我们需要做的是把每个绿色点在 5 左边还是右边弄出来，而且内部排序。

正解——排连通块

这一部分与子任务 1 类似。把粗青色边切掉，然后询问所有绿色点到当前蓝色点的蓝色后继（当前例子为 9）的距离。

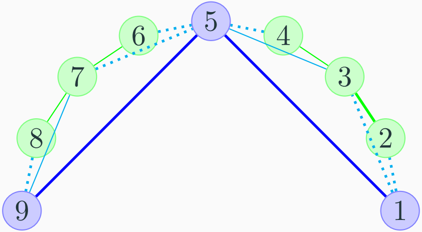


此时对于左边（$(5,9)$ 里）的绿点，到 9 的最短路长度构成起始为 1 的等差数列，越靠左距离越短；对于右边（$(1,5)$ 里）的绿点，到 9 的最短路长度构成起始为 3 的等差数列，越靠右距离越短。

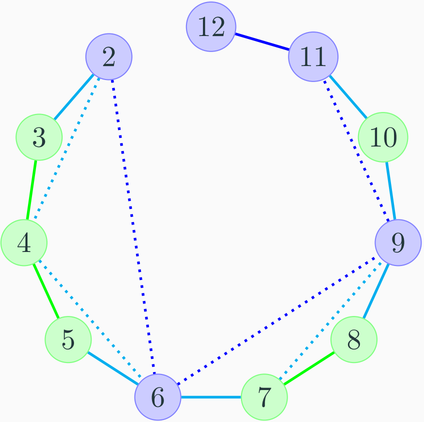
与加第二层的做法类似，容易在 $\lfloor 1.5m \rfloor$ 次询问内对 m 个绿点进行排序。

正解——排连通块

对于没有蓝色后继的情况（例如这里的 9），此时所有绿点都一定在它右边，所以另选一个点（例如这里的 7）来问就可以在 m 次询问内对 m 个绿点排序。

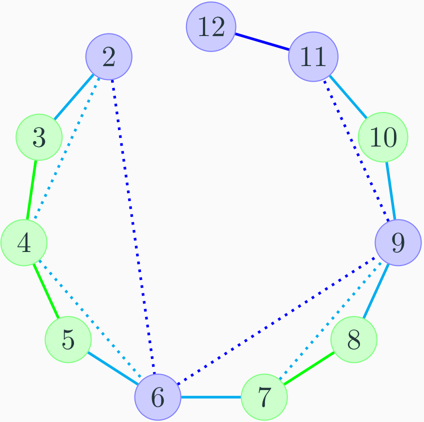


正解——继续加后面的层



找到所有的边之后，注意到第二层和第三层的点也可以串成一条链。我们把这条链保留，其他边删掉。

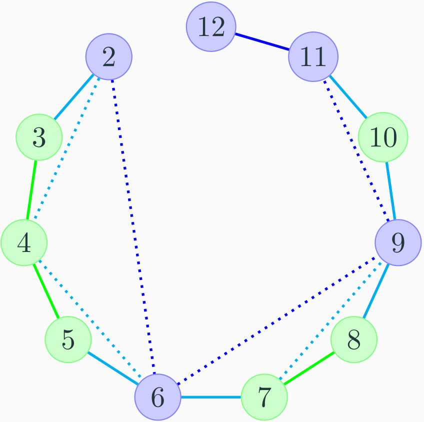
正解——继续加后面的层



找到所有的边之后，注意到第二层和第三层的点也可以串成一条链。我们把这条链保留，其他边删掉。

注意这条链上可能存在之前删掉的粗青色边，需要把它们找回来。

正解——继续加后面的层



找到所有的边之后，注意到第二层和第三层的点也可以串成一条链。我们把这条链保留，其他边删掉。

注意这条链上可能存在之前删掉的粗青色边，需要把它们找回来。

这条链上包含了所有的绿色边，而只有绿色边上可以继续往下接下面层的点，故重复以上过程即可。

正解——算一算操作次数

显然每一条边只会被恢复最多一次，所以恢复边的次数是对的。一个同样简单的分析可以得到 n 的上界，这是因为每个点只会对应一条粗青色边。

最短路的询问次数由以下几部分构成，假设第二层大小为 m:

- 初始分层 n;
- 第二层的排序 $\lceil 1.5m \rceil$;
- 第三层每个点找粗青色边 $(n-m)$;
- 第三层每个连通块排序 $\lfloor 1.5(n-m) \rfloor$。

因此总步数不超过 $3.5n$，可以通过。

完结撒花

~~画图画累死了~~