

图的联通性

2019 年 5 月 28 日

基础知识普及:

- 无向图的割点与割边
- 点双联通分量与边双联通分量
- 基础的最短路算法及其部分性质
- 有向图强联通分量 (貌似不是重点)

无向图联通性的一些定义

- (1) 割点: 满足删除这个点后, 图不联通的点 (集合).
- (2) 割边: 满足删除这条边后, 图不联通的边 (集合).
- (3) 点联通度: 满足删除若干个点后图不联通的最少需要的点的数量.
- (4) 边联通度: 满足删除若干条边后图不联通的最少需要的边的数量.

无向图的 Tarjan 算法

- Tarjan 算法基于对图的深度优先搜索, 并在每个节点引入两个新的值:
- $dfn[u]$: 结点 u 的时间戳, 即在 DFS 过程中 u 是第几个被访问的点.
- $low[u]$: 结点 u 通过非树边能够到达的 dfn 值最小的点.

无向图的 Tarjan 算法

- Tarjan 算法基于对图的深度优先搜索, 并在每个节点引入两个新的值:
- $dfn[u]$: 结点 u 的时间戳, 即在 DFS 过程中 u 是第几个被访问的点.
- $low[u]$: 结点 u 通过非树边能够到达的 dfn 值最小的点.
- 考虑每一条与 u 相连的边 (u, v) :
- 若 v 在 DFS 树上是结点 u 的子结点, 则更新 $low[u] \leftarrow \min(low[u], low[v])$.
- 否则, 更新 $low[u] \leftarrow \min(low[u], dfn[v])$.

无向图的 Tarjan 算法

- Tarjan 算法基于对图的深度优先搜索, 并在每个节点引入两个新的值:
- $dfn[u]$: 结点 u 的时间戳, 即在 DFS 过程中 u 是第几个被访问的点.
- $low[u]$: 结点 u 通过非树边能够到达的 dfn 值最小的点.
- 考虑每一条与 u 相连的边 (u, v) :
- 若 v 在 DFS 树上是结点 u 的子结点, 则更新 $low[u] \leftarrow \min(low[u], low[v])$.
- 否则, 更新 $low[u] \leftarrow \min(low[u], dfn[v])$.

这个算法在有向图上的过程差不多.

常用最短路算法的特殊应用

- Floyd 算法求最小环.
- Floyd 算法传递闭包.
- Dijkstra 算法求 k 短路.
- Spfa(Bellman Ford) 算法判断负权环.

例题选讲:

Knights of the Round Table

题意: 给出 N 个骑士之间 M 组相互仇恨的关系, 规定召开圆桌会议时相互仇恨的骑士不能坐在相邻的位置, 且总人数必须为奇数, 求有多少骑士不可能参加会议.

$$N \leq 1000, M \leq 1000000$$

分析: 发现实际上满足条件的一种开会方案就是原图补图的一个奇环, 同时我们发现, 只要一个边双联通分量中存在一个奇环, 那么这个边双中的所有点一定都在某个奇环上.

Knights of the Round Table

分析: 发现实际上满足条件的一种开会方案就是原图补图的一个奇环, 同时我们发现, 只要一个边双联通分量中存在一个奇环, 那么这个边双中的所有点一定都在某个奇环上.

接下来问题就变得简单了, 求出原图补图中的所有边双, 并用染色法判定二分图来确定是否有奇环.

思考题

题意: 给定一个无向联通图, 求添加最少的边使得这个图边双联通.

$$N, M \leq 1000000$$

分析: 首先, 我们不需要考虑所有的边双内部的联通情况, 那么我们可以把所有的边双缩点, 这样原图就变成了一棵无根树, 记 cnt 为这棵树中度为 1 的点的个数, 则边数为 $\lceil \frac{cnt}{2} \rceil$.

分析: 首先, 我们不需要考虑所有的边双内部的联通情况, 那么我们可以把所有的边双缩点, 这样原图就变成了一棵无根树, 记 cnt 为这棵树中度为 1 的点的个数, 则边数为 $\lceil \frac{cnt}{2} \rceil$.

如何构造满足条件的方案呢? 我们将所有这样的点两两配个对, 每次将 lca 最远的配对即可, 可能需要认真感受正确性什么的一下.

题意: 给出一个无向图 (边权为正整数), 多次询问删除一条边之后 S, T 之间的最短路长度.

$$N, M, Q \leq 200000$$

分析: 首先求出以 S, T 的单源最短路, 并建出最短路图, 图中包含所有满足 $dis_u + w(u, v) = dis_v$ 的边 (u, v) , 然后我们可以删除所有无法通过这样的边到达 T 的点, 根据最短路的性质我们可以知道这个新图缩完边双之后将形成一条链.

分析: 首先求出以 S, T 的单源最短路, 并建出最短路图, 图中包含所有满足 $dis_u + w(u, v) = dis_v$ 的边 (u, v) , 然后我们可以删除所有无法通过这样的边到达 T 的点, 根据最短路的性质我们可以知道这个新图缩完边双之后将形成一条链.

考虑删除的边, 发现如果这样的边不是新图的桥, 一定不会影响最短路的长度, 否则我们可以离线处理任意两个相邻的边双之间删除桥的答案:

分析: 首先求出以 S, T 的单源最短路, 并建出最短路图, 图中包含所有满足 $dis_u + w(u, v) = dis_v$ 的边 (u, v) , 然后我们可以删除所有无法通过这样的边到达 T 的点, 根据最短路的性质我们可以知道这个新图缩完边双之后将形成一条链.

考虑删除的边, 发现如果这样的边不是新图的桥, 一定不会影响最短路的长度, 否则我们可以离线处理任意两个相邻的边双之间删除桥的答案:

对每一条边 (u, v) 用 $disS_u + w(u, v) + disT_v$ 来表示它的值, 那么考虑所有不在新图上的边. 我们维护一个 set , 然后依次扫描每一个边双, 每次删除从之前的边双连向当前边双的边的值, 然后加入从这个边双到达后面的边双的值, 这时候删除这个边双与下一个边双之间的桥的答案就是 set 中的最小值.

题意: 给你一个无向联通图, 每次加入一条边并询问加入这条边之后图中桥的数量.

$$N \leq 100000, M \leq 200000, Q \leq 1000$$

分析: 在所有的边加入之前, 我们可以先求出原图的桥的数量, 这时候我们就可以忽略边双内部的边了: 我们将边双缩点, 会得到一棵树, 不难发现每次连接树上的两个点时, 这两个点到他们的 Lca 途中的所有边将被标记成非割边.

分析: 在所有的边加入之前, 我们可以先求出原图的桥的数量, 这时候我们就可以忽略边双内部的边了: 我们将边双缩点, 会得到一棵树, 不难发现每次连接树上的两个点时, 这两个点到他们的 Lca 途中的所有边将被标记成非割边.

其实这题可以暴力标记, 然后用并查集缩掉新加入的割边.

题意: 给定无向联通图, 多次询问两条边之间必经点的数量.

$$N, Q \leq 10000, M \leq 100000$$

分析: 两条边之间的必经点的数量与经过的路径上割点的数量相关, 为了方便统计, 我们可以先把将原图的点双联通分量缩点 (注意缩点的过程中是不考虑割点的),

分析: 两条边之间的必经点的数量与经过的路径上割点的数量相关, 为了方便统计, 我们可以先把将原图的点双联通分量缩点 (注意缩点的过程中是不考虑割点的),

然后我们可以得到一棵树, 这样问题就转化为求树上两点之间的割点数量了.

UR 2, 跳蚤公路

题意: 给出一个 N 个点, M 条边的有向图, 其中有三类边, 每次可以选择一个 x , 然后让所有的一类边权值加上 x , 所有的二类边的权值减去 x , 三类边不变. 对每一个点, 问从一号点出发到达这个点的路径上不存在负权环的 x 的取值有多少种.

$$N \leq 100 \quad M \leq 10000$$

UR 2, 跳蚤公路

分析: 考虑 Bellman Ford 算法的一个特殊性质: 记 $f[i][u]$ 表示从源点出发, 走不超过 i 步到达 u 的最短距离, 那么图中有负环当且仅当存在一个点 u , 满足 $f[n-1][u] > f[n][u]$.

UR 2, 跳蚤公路

分析: 考虑 Bellman Ford 算法的一个特殊性质: 记 $f[i][u]$ 表示从源点出发, 走不超过 i 步到达 u 的最短距离, 那么图中有负环当且仅当存在一个点 u , 满足 $f[n-1][u] > f[n][u]$.

在原问题中, 我们可以暴力找出每一个环, 计算这个环的系数, 然后解一个一次方程算出这个环的合法范围, 最后求交即可.

UR 2, 跳蚤公路

分析: 考虑 Bellman Ford 算法的一个特殊性质: 记 $f[i][u]$ 表示从源点出发, 走不超过 i 步到达 u 的最短距离, 那么图中有负环当且仅当存在一个点 u , 满足 $f[n-1][u] > f[n][u]$.

在原问题中, 我们可以暴力找出每一个环, 计算这个环的系数, 然后解一个一次方程算出这个环的合法范围, 最后求交即可.

有了这个性质, 我们可以加以改进, 记 $f[i][u][s]$ 表示从源点出发, 走不超过 i 步, 到达 u 且经过的边的系数和恰好为 s 的最短距离, 显然这时就要求满足:

$$\min\{ix + f[n-1][u][i]\} \leq \min\{jx + f[n][u][j]\}$$

把每个环的答案算出来, 然后更新能够到达的点的 ans 即可.