Tarjan 算法及其应用

Tarjan 算法 · 缩点 · 2-SAT 问题

R9M9Q6

Jul 21, 2022

章节目录

概念补充 · 无向图

连通

若图上两点 u, v 之间存在路径，那么称这两个点是**连通**的。

概念补充 · 无向图

连通

若图上两点 u, v 之间存在路径，那么称这两个点是**连通**的。

连通分量

一张无向图的每个**极大**连通块是这张无向图的**连通分量**。
注：这一部分提到的所有叫“XX 分量”的都是极大的满足某一条件的子图。

概念补充 · 无向图

连通

若图上两点 u, v 之间存在路径，那么称这两个点是**连通**的。

连通分量

一张无向图的每个**极大**连通块是这张无向图的**连通分量**。
注：这一部分提到的所有叫“XX 分量”的都是极大的满足某一条件的子图。

点双连通分量（点双）

一张无向图的子图中满足“其中的任意两点之间都有至少两条不同时经过除了这两点以外的点的简单路径”的极大子图称为这张图的**点双连通分量**。

概念补充 · 无向图

连通

若图上两点 u, v 之间存在路径，那么称这两个点是**连通**的。

连通分量

一张无向图的每个**极大**连通块是这张无向图的**连通分量**。
注：这一部分提到的所有叫“XX 分量”的都是极大的满足某一条件的子图。

点双连通分量（点双）

一张无向图的子图中满足“其中的任意两点之间都有至少两条不同时经过除了这两点以外的点的简单路径”的极大子图称为这张图的**点双连通分量**。
特别地，一条边及其两端点在**单独存在**时构成一个点双连通分量。

概念补充 · 无向图

连通

若图上两点 u, v 之间存在路径，那么称这两个点是**连通**的。

连通分量

一张无向图的每个**极大**连通块是这张无向图的**连通分量**。
注：这一部分提到的所有叫“XX 分量”的都是极大的满足某一条件的子图。

点双连通分量（点双）

一张无向图的子图中满足“其中的任意两点之间都有至少两条不同时经过除了这两点以外的点的简单路径”的极大子图称为这张图的**点双连通分量**。
特别地，一条边及其两端点在**单独存在**时构成一个点双连通分量。
若整张图的点双连通分量就是本身，那么这张图就是一张**点双连通图**。

概念补充 · 无向图

边双连通分量（边双）

一张无向图的子图中满足“其中的任意两点之间都有至少两条不同时经过同一条边的简单路径”的极大子图称为这张图的**边双连通分量**。

概念补充 · 无向图

边双连通分量（边双）

一张无向图的子图中满足“其中的任意两点之间都有至少两条不同时经过同一条边的简单路径”的极大子图称为这张图的**边双连通分量**。
若整张图的边双连通分量就是本身，那么这张图就是一张**边双连通图**。

概念补充 · 无向图

边双连通分量（边双）

一张无向图的子图中满足“其中的任意两点之间都有至少两条不同时经过同一条边的简单路径”的极大子图称为这张图的**边双连通分量**。
若整张图的边双连通分量就是本身，那么这张图就是一张**边双连通图**。

割点（割顶）

在一张无向图中，若删去某个点及所有与这个点相连的边后无向图的连通分量增加，则称这个点为这张图的**割点（割顶）**。

概念补充 · 无向图

边双连通分量（边双）

一张无向图的子图中满足“其中的任意两点之间都有至少两条不同时经过同一条边的简单路径”的极大子图称为这张图的**边双连通分量**。
若整张图的边双连通分量就是本身，那么这张图就是一张**边双连通图**。

割点（割顶）

在一张无向图中，若删去某个点及所有与这个点相连的边后无向图的连通分量增加，则称这个点为这张图的**割点（割顶）**。

割边（桥）

在一张无向图中，若删去某条边后无向图的连通分量增加，则称这条边为这张图的**割边（桥）**。

概念补充 · 有向图

强连通

对于一张有向图，若图上两点 u, v 都满足存在从 u 到 v 和从 v 到 u 的路径，那么称这两个点是**强连通**的。

概念补充 · 有向图

强连通

对于一张有向图，若图上两点 u, v 都满足存在从 u 到 v 和从 v 到 u 的路径，那么称这两个点是**强连通**的。

强连通图

对于一张有向图，若图上任意两点 u, v 都是强连通的，那么称这张图为一张**强连通图**。

概念补充 · 有向图

强连通

对于一张有向图，若图上两点 u, v 都满足存在从 u 到 v 和从 v 到 u 的路径，那么称这两个点是**强连通**的。

强连通图

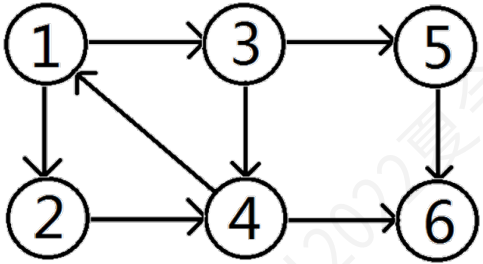
对于一张有向图，若图上任意两点 u, v 都是强连通的，那么称这张图为一张**强连通图**。

强连通分量

一张有向图极大强连通子图称为这张图的一个**强连通分量**。

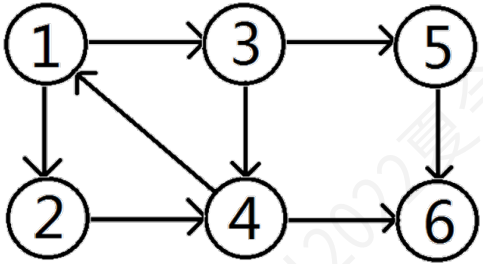
小测验 1.1

写出下图中的强连通分量：



小测验 1.1

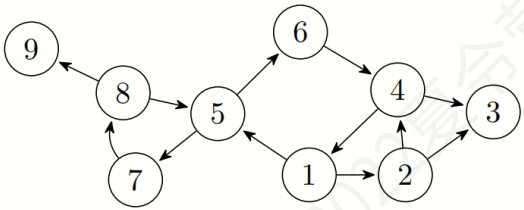
写出下图中的强连通分量：



$\{1,2,3,4\},\{5\},\{6\}$。

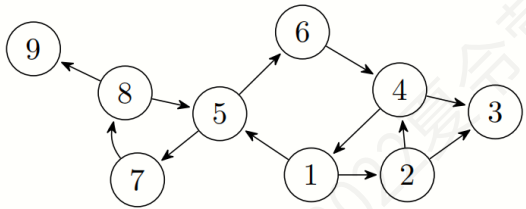
小测验 1.2

写出下图中的强连通分量：



小测验 1.2

写出下图中的强连通分量：



$\{1,2,4,5,6,7,8\},\{3\},\{9\}$。

概念补充 · 有向图

DFS 树

从图上的某一个点（即之后 DFS 树的根）出发对整张图做 DFS，这个过程中经过的边和原图上所有的点构成的树形结构称为 DFS 树。

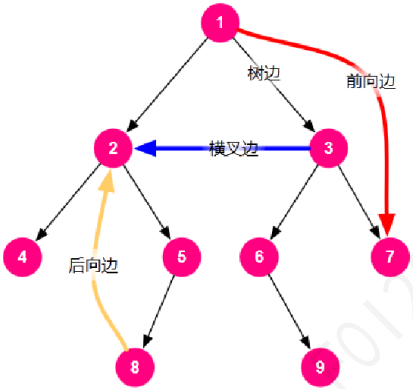
概念补充 · 有向图

DFS 树

从图上的某一个点（即之后 DFS 树的根）出发对整张图做 DFS，这个过程中经过的边和原图上所有的点构成的树形结构称为 DFS 树。

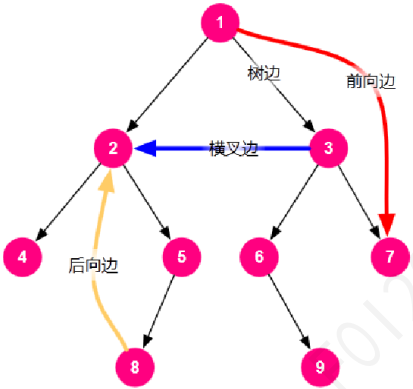
特别地，定义一个强连通分量的根为这个强连通分量在 DFS 树中深度最浅的一个点。

边的分类



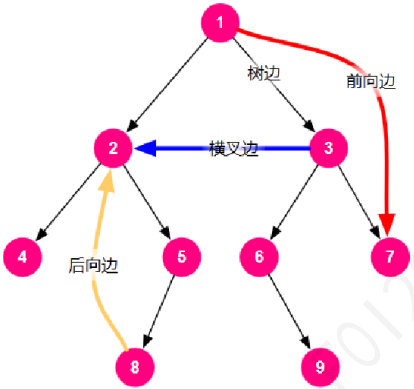
从一个点开始对有向图做 DFS 后可以得到一棵 DFS 树，树上的边就称为**树边**。

边的分类



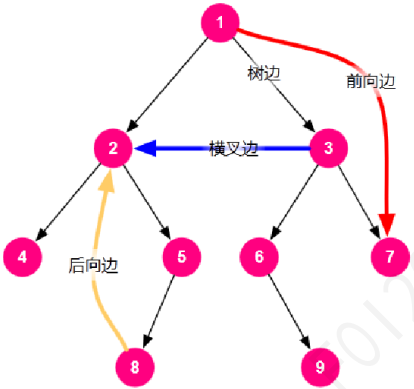
从一个点开始对有向图做 DFS 后可以得到一棵 DFS 树，树上的边就称为**树边**。
从一个点指向其子树中的点的边称为**前向边**。

边的分类



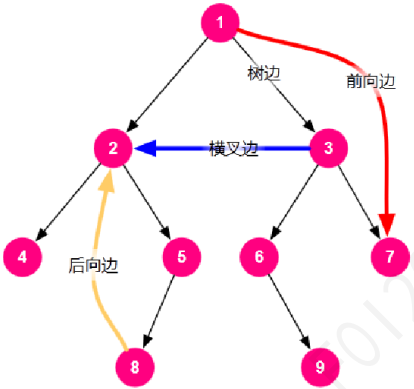
从一个点开始对有向图做 DFS 后可以得到一棵 DFS 树，树上的边就称为**树边**。
从一个点指向其子树中的点的边称为**前向边**。
从一个点指向其祖先的边称为**后向边（返祖边）**。

边的分类



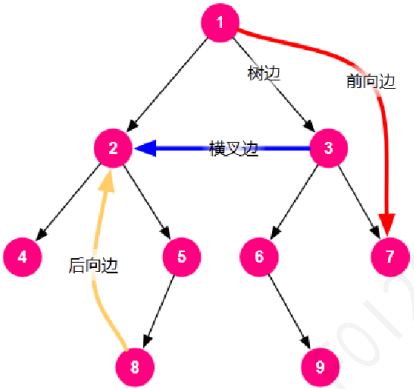
从一个点开始对有向图做 DFS 后可以得到一棵 DFS 树，树上的边就称为**树边**。
从一个点指向其子树中的点的边称为**前向边**。
从一个点指向其祖先的边称为**后向边（返祖边）**。
若边的两端点位于它们的 LCA 的两个不同子树中，那么这条边是一条**横叉边**。横叉边满足起点被遍历到的时间晚于终点（否则这条边会变成树边）。

边的分类



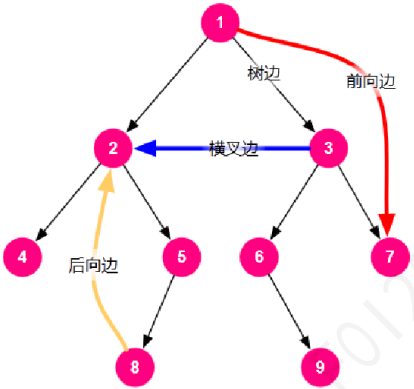
从一个点开始对有向图做 DFS 后可以得到一棵 DFS 树，树上的边就称为**树边**。
从一个点指向其子树中的点的边称为**前向边**。
从一个点指向其祖先的边称为**后向边（返祖边）**。
若边的两端点位于它们的 LCA 的两个不同子树中，那么这条边是一条**横叉边**。横叉边满足起点被遍历到的时间晚于终点（否则这条边会变成树边）。
注：边的分类是由 DFS 的情况决定的，而不是固有的。

边的分类



从一个点开始对有向图做 DFS 后可以得到一棵 DFS 树，树上的边就称为**树边**。
从一个点指向其子树中的点的边称为**前向边**。
从一个点指向其祖先的边称为**后向边（返祖边）**。
若边的两端点位于它们的 LCA 的两个不同子树中，那么这条边是一条**横叉边**。横叉边满足起点被遍历到的时间晚于终点（否则这条边会变成树边）。
注：边的分类是由 DFS 的情况决定的，而不是固有的。
对于无向图，类比上面的定义，那么会有哪几种边?

边的分类



从一个点开始对有向图做 DFS 后可以得到一棵 DFS 树，树上的边就称为**树边**。
从一个点指向其子树中的点的边称为**前向边**。
从一个点指向其祖先的边称为**后向边（返祖边）**。
若边的两端点位于它们的 LCA 的两个不同子树中，那么这条边是一条**横叉边**。横叉边满足起点被遍历到的时间晚于终点（否则这条边会变成树边）。
注：边的分类是由 DFS 的情况决定的，而不是固有的。
对于无向图，类比上面的定义，那么会有哪几种边？
树边、返祖边（不区分前向边与后向边）。

Tarjan 算法 · 信息维护

Tarjan 算法在求解过程中主要维护以下三个信息：

Tarjan 算法 · 信息维护

Tarjan 算法在求解过程中主要维护以下三个信息：
1. 一个栈，用于记录当前已经访问过但是还没有被归类到任一强连通分量的节点。

Tarjan 算法 · 信息维护

Tarjan 算法在求解过程中主要维护以下三个信息：

1. 一个栈，用于记录当前已经访问过但是还没有被归类到任一强连通分量的节点。
2. $dfn[u]$：表示点 u 的 DFS 序（第一次出现的时间），即标记点 u 是第几个被 DFS 到的点。

Tarjan 算法 · 信息维护

Tarjan 算法在求解过程中主要维护以下三个信息：

1. 一个栈，用于记录当前已经访问过但是还没有被归类到任一强连通分量的节点。

2. $dfn[u]$：表示点 u 的 DFS 序（第一次出现的时间），即标记点 u 是第几个被 DFS 到的点。

3. $low[u]$：从 u 或者以 u 为根的子树中的节点出发通过一条反祖边或者横叉边可以到达的时间戳最小的，且能够到达 u 的结点 v 的时间戳。

Tarjan 算法 · low 的计算

假设当前正在对点 u 做 DFS 且正在处理一条从 u 到 v 的边。

Tarjan 算法 · low 的计算

假设当前正在对点 u 做 DFS 且正在处理一条从 u 到 v 的边。

1. 若点 v 还没有被搜索过，那么这条边为树边，此时有 $low[u] = \min\{low[u], low[v]\}$，即将这一次走非树边的机会留给子树。

Tarjan 算法 · low 的计算

假设当前正在对点 u 做 DFS 且正在处理一条从 u 到 v 的边。

1. 若点 v 还没有被搜索过，那么这条边为树边，此时有 $low[u] = \min\{low[u], low[v]\}$，即将这一次走非树边的机会留给子树。

2. 若这条边是一条返祖边（一定满足条件）或满足条件的横叉边，那么有 $low[u] = \min\{low[u], dfn[v]\}$，将这一次走非树边的机会用掉。

Tarjan 算法 · low 的计算

假设当前正在对点 u 做 DFS 且正在处理一条从 u 到 v 的边。

1. 若点 v 还没有被搜索过，那么这条边为树边，此时有 $low[u] = \min\{low[u], low[v]\}$，即将这一次走非树边的机会留给子树。

2. 若这条边是一条返祖边（一定满足条件）或满足条件的横叉边，那么有 $low[u] = \min\{low[u], dfn[v]\}$，将这一次走非树边的机会用掉。

3. 若这条边是一条前向边，那么根据定义，它对 $low[u]$ 的计算是没有影响的。

Tarjan 算法 · 栈的维护

强连通分量的根的判定

节点 u 是强连通分量的根等价于 $dfn[u]$ 和 $low[u]$ 相等。

Tarjan 算法 · 栈的维护

强连通分量的根的判定

节点 u 是强连通分量的根等价于 $dfn[u]$ 和 $low[u]$ 相等。

在 DFS 的过程中，每结束对一个节点的搜索就判断一下这个节点的 low 与 dfn 的大小关系。如果这个节点出现了 low 与 dfn 相等的情况，那么这个点就是强连通分量的根，要将整个强连通分量从栈中弹出。由于 dfn 的子树连续性（一个子树里面的节点的 dfn 是连续的），所以可以不断执行出栈操作直到强连通分量的根出栈为止。

Tarjan 算法 · 栈的维护

强连通分量的根的判定

节点 u 是强连通分量的根等价于 $dfn[u]$ 和 $low[u]$ 相等。

在 DFS 的过程中，每结束对一个节点的搜索就判断一下这个节点的 low 与 dfn 的大小关系。如果这个节点出现了 low 与 dfn 相等的情况，那么这个点就是强连通分量的根，要将整个强连通分量从栈中弹出。由于 dfn 的子树连续性（一个子树里面的节点的 dfn 是连续的），所以可以不断执行出栈操作直到强连通分量的根出栈为止。

最后剩下一个问题，如何判断正在搜索的返祖边或横叉边是否符合条件呢?

Tarjan 算法 · 栈的维护

强连通分量的根的判定

节点 u 是强连通分量的根等价于 $dfn[u]$ 和 $low[u]$ 相等。

在 DFS 的过程中，每结束对一个节点的搜索就判断一下这个节点的 low 与 dfn 的大小关系。如果这个节点出现了 low 与 dfn 相等的情况，那么这个点就是强连通分量的根，要将整个强连通分量从栈中弹出。由于 dfn 的子树连续性（一个子树里面的节点的 dfn 是连续的），所以可以不断执行出栈操作直到强连通分量的根出栈为止。

最后剩下一个问题，如何判断正在搜索的返祖边或横叉边是否符合条件呢?

记 $L = LCA(u, v)$，如果点 v 还在栈里，那么 v 或其子节点必然存在指向 L 或其祖先的边，故边符合条件等价于 v 在栈里。

割点、割边的判定

割点的判定

在 DFS 树中，如果以某个点为根的子树中没有向外（不包括根的父节点）的边，那么它的**父节点**就是一个割点。在算法中表现为 $dfn[u] \leq low[v]$（注意**可以取等**）。
特别注意：当根节点只有一个子树时，根节点不是割点。

割点、割边的判定

割点的判定

在 DFS 树中，如果以某个点为根的子树中没有向外（不包括根的父节点）的边，那么它的**父节点**就是一个割点。在算法中表现为 $dfn[u] \leq low[v]$（注意**可以取等**）。
特别注意：当根节点只有一个子树时，根节点不是割点。

割边的判定

在 DFS 树中，如果以某个点为根的子树中没有向外的边，那么从它到它的父节点的边就是一条割边。在算法中表现为 $dfn[u] < low[v]$（注意**不能取等**）。

「UOJ 67」新年的毒瘤

给出一张 n 个点 m 条边的无向图，问有哪些结点满足：删去该点后原图变为一棵树。
数据范围：$n, m \leq 10^5$。

「UOJ 67」新年的毒瘤

给出一张 n 个点 m 条边的无向图，问有哪些结点满足：删去该点后原图变为一棵树。

数据范围：$n, m \leq 10^5$。

从定义出发，树是边数比点数少 1 的连通图。前一个条件要求删完之后剩下的图应该为 $n-1$ 个点 $n-2$ 条边的图，后一个条件要求这个点不是割点。用这两个条件筛出来的点就是符合条件的点。

时间复杂度（同 Tarjan）:$\mathbb{O}(m+n)$。

「POI 2008」Blockade

给出一张 n 个点 m 条边的无向图，对于每个点分别求出将它删去后图上不连通点对个数的变化量。

数据范围：$1 \leq n \leq 10^5, 1 \leq m \leq 5 \times 10^5$。

「POI 2008」Blockade

给出一张 n 个点 m 条边的无向图，对于每个点分别求出将它删去后图上不连通点对个数的变化量。

数据范围：$1 \leq n \leq 10^5, 1 \leq m \leq 5 \times 10^5$。

根据定义，如果删的点是割点，才会导致不连通点对数目增加。对于涉及到割点的问题，一般要分根节点和非根节点两类讨论。

「POI 2008」Blockade

给出一张 n 个点 m 条边的无向图，对于每个点分别求出将它删去后图上不连通点对个数的变化量。

数据范围：$1 \leq n \leq 10^5, 1 \leq m \leq 5 \times 10^5$。

根据定义，如果删的点是割点，才会导致不连通点对数目增加。对于涉及到割点的问题，一般要分根节点和非根节点两类讨论。

对于根节点，如果它是一个割点，那么将它删除后不同子树之间节点变得不连通。若各子树的大小分别为 $a_1, a_2, \cdots, a_t$，则由乘法原理，变化量：

$$\delta = \sum_{1 \leq i < j \leq t} a_i a_j = \frac{1}{2}\left(\left(\sum_{i=1}^{t} a_i\right)^2 - \left(\sum_{i=1}^{t} a_i^2\right)\right)$$

「POI 2008」Blockade

给出一张 n 个点 m 条边的无向图，对于每个点分别求出将它删去后图上不连通点对个数的变化量。

数据范围：$1 \le n \le 10^5, 1 \le m \le 5 \times 10^5$。

根据定义，如果删的点是割点，才会导致不连通点对数目增加。对于涉及到割点的问题，一般要分根节点和非根节点两类讨论。

对于根节点，如果它是一个割点，那么将它删除后不同子树之间节点变得不连通。若各子树的大小分别为 $a_1, a_2, \cdots, a_t$，则由乘法原理，变化量：

$$\delta = \sum_{1 \le i < j \le t} a_i a_j = \frac{1}{2}\left(\left(\sum_{i=1}^{t} a_i\right)^2 - \left(\sum_{i=1}^{t} a_i^2\right)\right)$$

对于非根节点，将它删去后，并非所有的子树都会变成独立的连通分量。只有当子树满足 $low[v] \ge dfn[u]$ 时，子树中没有向外指的边，此时在删去点 u 后，以 v 为根的子树就会变成独立的连通分量。在完成判定之后再用类似的步骤按照乘法原理即可得出答案。

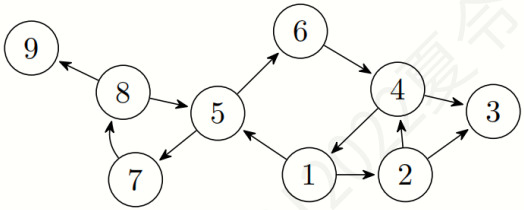
章节目录

有向图缩点

缩点是一种基于 Tarjan 的处理有向图的算法，其核心是在用 Tarjan 计算出图上的强连通分量后，建立一个新的图，将每个强连通分量变成一个点存在新图中；在原图的所有边中，只有原图中两端位于不同强连通分量的边保留下来，在新图中连接两端点对应的强连通分量对应的新点且方向不变。

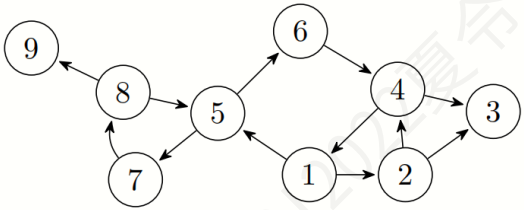
有向图缩点

缩点是一种基于 Tarjan 的处理有向图的算法，其核心是在用 Tarjan 计算出图上的强连通分量后，建立一个新的图，将每个强连通分量变成一个点存在新图中；在原图的所有边中，只有原图中两端位于不同强连通分量的边保留下来，在新图中连接两端点对应的强连通分量对应的新点且方向不变。



有向图缩点

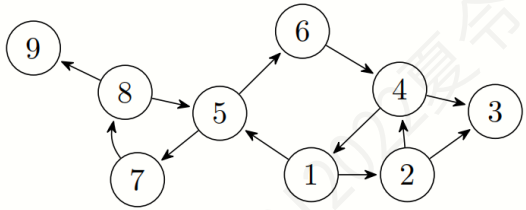
缩点是一种基于 Tarjan 的处理有向图的算法，其核心是在用 Tarjan 计算出图上的强连通分量后，建立一个新的图，将每个强连通分量变成一个点存在新图中；在原图的所有边中，只有原图中两端位于不同强连通分量的边保留下来，在新图中连接两端点对应的强连通分量对应的新点且方向不变。



为什么要缩点?

有向图缩点

缩点是一种基于 Tarjan 的处理有向图的算法，其核心是在用 Tarjan 计算出图上的强连通分量后，建立一个新的图，将每个强连通分量变成一个点存在新图中；在原图的所有边中，只有原图中两端位于不同强连通分量的边保留下来，在新图中连接两端点对应的强连通分量对应的新点且方向不变。



为什么要缩点？缩点后得到的新图是一张有向无环图（DAG），这就意味着在上面直接进行搜索不会进入死循环，而拓扑序遍历、拓扑图 DP 等算法也可以在上面使用。

「洛谷 2002」消息扩散

有 n 个城市，中间有 m 条单向道路连接，消息会沿着道路扩散，现在给出 n 个城市及其之间的道路，问至少需要在几个城市发布消息才能让这所有 n 个城市都得到消息。

数据范围：$1 \le n \le 10^5, 1 \le m \le 5 \times 10^5$。

「洛谷 2002」消息扩散

有 n 个城市，中间有 m 条单向道路连接，消息会沿着道路扩散，现在给出 n 个城市及其之间的道路，问至少需要在几个城市发布消息才能让这所有 n 个城市都得到消息。

数据范围：$1 \le n \le 10^5, 1 \le m \le 5 \times 10^5$。

在一个强连通分量中的任意一个城市发布消息后，整个强连通分量里的城市都会收到消息，所以在强连通分量中具体那个点发布消息是不重要的，即整张图具备缩点的条件。

「洛谷 2002」消息扩散

有 n 个城市，中间有 m 条单向道路连接，消息会沿着道路扩散，现在给出 n 个城市及其之间的道路，问至少需要在几个城市发布消息才能让这所有 n 个城市都得到消息。

数据范围：$1 \le n \le 10^5, 1 \le m \le 5 \times 10^5$。

在一个强连通分量中的任意一个城市发布消息后，整个强连通分量里的城市都会收到消息，所以在强连通分量中具体那个点发布消息是不重要的，即整张图具备缩点的条件。

在缩点之后，DAG 中入度为 0 的点不能通过其它的点获得消息，其它点则可以从指向它的点获得消息。所以答案即为 DAG 中入度为 0 的点的数量。

「APIO 2009」抢掠计划

给出一张有向图，试求出从 1 号点到若干个指定的点的最长路径（经过点最多的路径，重复经过算一次）。

数据范围：$n, m \leq 5 \times 10^5$。

「APIO 2009」抢掠计划

给出一张有向图，试求出从 1 号点到若干个指定的点的最长路径（经过点最多的路径，重复经过算一次）。

数据范围：$n, m \leq 5 \times 10^5$。

由于点可以重复走且强连通分量内部任意两点双向可达，所以每到一个强连通分量后的最优策略就是将这个强连通分量中的点全部走遍。

「APIO 2009」抢掠计划

给出一张有向图，试求出从 1 号点到若干个指定的点的最长路径（经过点最多的路径，重复经过算一次）。

数据范围：$n, m \leq 5 \times 10^5$。

由于点可以重复走且强连通分量内部任意两点双向可达，所以每到一个强连通分量后的最优策略就是将这个强连通分量中的点全部走遍。

于是我们将图缩点得到一个 DAG，令每一个新点的权值为它在原图中对应的强连通分量的点的个数。那么问题转化为在 DAG 上走一条点权值和最大的路径。

「APIO 2009」抢掠计划

给出一张有向图，试求出从 1 号点到若干个指定的点的最长路径（经过点最多的路径，重复经过算一次）。

数据范围：$n, m \leq 5 \times 10^5$。

由于点可以重复走且强连通分量内部任意两点双向可达，所以每到一个强连通分量后的最优策略就是将这个强连通分量中的点全部走遍。

于是我们将图缩点得到一个 DAG，令每一个新点的权值为它在原图中对应的强连通分量的点的个数。那么问题转化为在 DAG 上走一条点权值和最大的路径。

由于此时图上已经没有环了，可用 SPFA 或拓扑序 DP 等方式求出到每个点的最大权值路径。

「USACO 2015 Jan.」Grass Cownoisseur

给定一张有向图，至多可以将一条边反向，要求选一条起点和终点都为 1 的路径，求最多可以经过几个点（重复经过算一次）。

数据范围：$n, m \leq 5 \times 10^5$。

「USACO 2015 Jan.」Grass Cownoisseur

给定一张有向图，至多可以将一条边反向，要求选一条起点和终点都为 1 的路径，求最多可以经过几个点（重复经过算一次）。

数据范围：$n, m \leq 5 \times 10^5$。

和上一题一样，每到一个强连通分量后的最优策略是将这个强连通分量中的点全部走遍。

「USACO 2015 Jan.」Grass Cownoisseur

给定一张有向图，至多可以将一条边反向，要求选一条起点和终点都为 1 的路径，求最多可以经过几个点（重复经过算一次）。

数据范围：$n, m \leq 5 \times 10^5$。

和上一题一样，每到一个强连通分量后的最优策略是将这个强连通分量中的点全部走遍。

所以我们将图缩点得到一个 DAG 并定义每个点的权值为原图上强连通分量中的点的个数。对于 DAG 上的每个点 u，预处理出 $1 \to u$ 和 $u \to 1$ 的最大权值路径。之后枚举每一条边的反向的情况并比较答案即可。

「USACO 5.3」Network of Schools

给出一张 n 个点 m 条边的有向图，求至少添加几条边，使得从任意一个点出发，都能遍历整张图。

数据范围：$2 \leq n \leq 10^5$。

「USACO 5.3」Network of Schools

给出一张 n 个点 m 条边的有向图，求至少添加几条边，使得从任意一个点出发，都能遍历整张图。

数据范围：$2 \leq n \leq 10^5$。

不说人话版本：问最少加几条边之后整个图能变成一整个强连通分量。

「USACO 5.3」Network of Schools

给出一张 n 个点 m 条边的有向图，求至少添加几条边，使得从任意一个点出发，都能遍历整张图。

数据范围：$2 \leq n \leq 10^5$。

不说人话版本：问最少加几条边之后整个图能变成一整个强连通分量。

由于加边不会破坏原有的强连通分量，所以可以将所有的强连通分量缩点来简化问题。此时问题变为：给出一张 DAG，问最少加几条边使得它能变成一个强连通分量。

「USACO 5.3」Network of Schools

给出一张 n 个点 m 条边的有向图，求至少添加几条边，使得从任意一个点出发，都能遍历整张图。

数据范围：$2 \leq n \leq 10^5$。

不说人话版本：问最少加几条边之后整个图能变成一整个强连通分量。

由于加边不会破坏原有的强连通分量，所以可以将所有的强连通分量缩点来简化问题。此时问题变为：给出一张 DAG，问最少加几条边使得它能变成一个强连通分量。

能遍历整张图的要求会在 DAG 的哪个地方无法实现？

「USACO 5.3」Network of Schools

给出一张 n 个点 m 条边的有向图，求至少添加几条边，使得从任意一个点出发，都能遍历整张图。

数据范围：$2 \leq n \leq 10^5$。

不说人话版本：问最少加几条边之后整个图能变成一整个强连通分量。

由于加边不会破坏原有的强连通分量，所以可以将所有的强连通分量缩点来简化问题。此时问题变为：给出一张 DAG，问最少加几条边使得它能变成一个强连通分量。

能遍历整张图的要求会在 DAG 的哪个地方无法实现？对于出度为 0 的点，它不能到达别的点；对于入度为 0 的点，别的点不能到它。只要这两类特殊的点的连通性问题被解决了，其它的点的问题也自然能解决。

「USACO 5.3」Network of Schools

给出一张 n 个点 m 条边的有向图，求至少添加几条边，使得从任意一个点出发，都能遍历整张图。

数据范围：$2 \leq n \leq 10^5$。

不说人话版本：问最少加几条边之后整个图能变成一整个强连通分量。

由于加边不会破坏原有的强连通分量，所以可以将所有的强连通分量缩点来简化问题。此时问题变为：给出一张 DAG，问最少加几条边使得它能变成一个强连通分量。

能遍历整张图的要求会在 DAG 的哪个地方无法实现？对于出度为 0 的点，它不能到达别的点；对于入度为 0 的点，别的点不能到它。只要这两类特殊的点的连通性问题被解决了，其它的点的问题也自然能解决。

于是问题又转化为，至少加多少条边，使得 DAG 中没有入度或出度为 0 的点。

「USACO 5.3」Network of Schools

给出一张 n 个点 m 条边的有向图，求至少添加几条边，使得从任意一个点出发，都能遍历整张图。

数据范围：$2 \le n \le 10^5$。

不说人话版本：问最少加几条边之后整个图能变成一整个强连通分量。

由于加边不会破坏原有的强连通分量，所以可以将所有的强连通分量缩点来简化问题。此时问题变为：给出一张 DAG，问最少加几条边使得它能变成一个强连通分量。

能遍历整张图的要求会在 DAG 的哪个地方无法实现？对于出度为 0 的点，它不能到达别的点；对于入度为 0 的点，别的点不能到它。只要这两类特殊的点的连通性问题被解决了，其它的点的问题也自然能解决。

于是问题又转化为，至少加多少条边，使得 DAG 中没有入度或出度为 0 的点。

此时最优策略就是让每个入度为 0 的点和出度为 0 的点连边配对，故答案为（缩点后 DAG 中）max{ 入度为 0 的点的个数，出度为 0 的点的个数 }。

有向图缩点小结

什么情况下可以缩点?

有向图缩点小结

什么情况下可以缩点?

1. 某一个点对答案的贡献与它本身在强连通分量中的位置关系不大，而与它在哪个强连通分量有关。

有向图缩点小结

什么情况下可以缩点?

1. 某一个点对答案的贡献与它本身在强连通分量中的位置关系不大，而与它在哪个强连通分量有关。

2. 涉及到强连通分量的连通性性质（两两双向可达、从一个点出发可以走遍整个强连通分量等）。

有向图缩点小结

什么情况下可以缩点?

1. 某一个点对答案的贡献与它本身在强连通分量中的位置关系不大，而与它在哪个强连通分量有关。

2. 涉及到强连通分量的连通性性质（两两双向可达、从一个点出发可以走遍整个强连通分量等）。

什么情况下需要缩点?

有向图缩点小结

什么情况下可以缩点?

1. 某一个点对答案的贡献与它本身在强连通分量中的位置关系不大，而与它在哪个强连通分量有关。

2. 涉及到强连通分量的连通性性质（两两双向可达、从一个点出发可以走遍整个强连通分量等）。

什么情况下需要缩点?

1. 要进行搜索，且不缩点会导致死循环或更复杂的讨论。

有向图缩点小结

什么情况下可以缩点?

1. 某一个点对答案的贡献与它本身在强连通分量中的位置关系不大，而与它在哪个强连通分量有关。

2. 涉及到强连通分量的连通性性质（两两双向可达、从一个点出发可以走遍整个强连通分量等）。

什么情况下需要缩点?

1. 要进行搜索，且不缩点会导致死循环或更复杂的讨论。

2. 需要对图进行 DP，为了不出现循环，需要按照拓扑序列进行，此时图上不能有强连通分量存在。

有向图缩点小结

什么情况下可以缩点?

1. 某一个点对答案的贡献与它本身在强连通分量中的位置关系不大，而与它在哪个强连通分量有关。

2. 涉及到强连通分量的连通性性质（两两双向可达、从一个点出发可以走遍整个强连通分量等）。

什么情况下需要缩点?

1. 要进行搜索，且不缩点会导致死循环或更复杂的讨论。

2. 需要对图进行 DP，为了不出现循环，需要按照拓扑序列进行，此时图上不能有强连通分量存在。

3. 对于一些特殊情况，缩点还可以减少点和边的数量，降低时间复杂度。

无向图缩点

在无向图中，与强连通相对应的概念是点双和边双。类似地，我们在处理一些无向图的问题的时候也可以将点双和边双缩成一个点。

无向图缩点

在无向图中，与强连通相对应的概念是点双和边双。类似地，我们在处理一些无向图的问题的时候也可以将点双和边双缩成一个点。

对于缩边双连通分量的情况，若点 u 满足 $low[u] = dfn[u]$（类比强连通分量），那么 DFS 树中以 u 为根的子树中还在栈里的节点会构成一个边双，此时不断执行出栈操作直至点 u 出栈即可提取出这个边双中的点，之后将每个边双中的点看成一个新的点，边双之间的连边则保留。

无向图缩点

在无向图中，与强连通相对应的概念是点双和边双。类似地，我们在处理一些无向图的问题的时候也可以将点双和边双缩成一个点。

对于缩边双连通分量的情况，若点 u 满足 $low[u] = dfn[u]$（类比强连通分量），那么 DFS 树中以 u 为根的子树中还在栈里的节点会构成一个边双，此时不断执行出栈操作直至点 u 出栈即可提取出这个边双中的点，之后将每个边双中的点看成一个新的点，边双之间的连边则保留。

对于缩点双连通分量的情况，若点 u 及其子节点 v 满足 $low[v] \geq dfn[u]$（类比割点的情况），那么 DFS 树中以 u 为根的子树中还在栈里的节点会构成一个点双，此时不断执行出栈操作直至点 u 出栈即可提取出这个点双中的点。

无向图缩点

在无向图中，与强连通相对应的概念是点双和边双。类似地，我们在处理一些无向图的问题的时候也可以将点双和边双缩成一个点。

对于缩边双连通分量的情况，若点 u 满足 $low[u] = dfn[u]$（类比强连通分量），那么 DFS 树中以 u 为根的子树中还在栈里的节点会构成一个边双，此时不断执行出栈操作直至点 u 出栈即可提取出这个边双中的点，之后将每个边双中的点看成一个新的点，边双之间的连边则保留。

对于缩点双连通分量的情况，若点 u 及其子节点 v 满足 $low[v] \geq dfn[u]$（类比割点的情况），那么 DFS 树中以 u 为根的子树中还在栈里的节点会构成一个点双，此时不断执行出栈操作直至点 u 出栈即可提取出这个点双中的点。不过，由于不同点双之间存在点的共用，点双的缩点在连边时不能像之前一样直接保留部分的边。

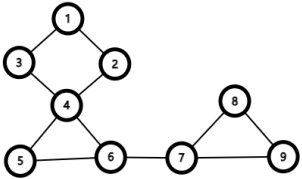
无向图缩点

在无向图中，与强连通相对应的概念是点双和边双。类似地，我们在处理一些无向图的问题的时候也可以将点双和边双缩成一个点。

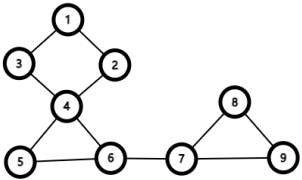
对于缩边双连通分量的情况，若点 u 满足 $low[u] = dfn[u]$（类比强连通分量），那么 DFS 树中以 u 为根的子树中还在栈里的节点会构成一个边双，此时不断执行出栈操作直至点 u 出栈即可提取出这个边双中的点，之后将每个边双中的点看成一个新的点，边双之间的连边则保留。

对于缩点双连通分量的情况，若点 u 及其子节点 v 满足 $low[v] \geq dfn[u]$（类比割点的情况），那么 DFS 树中以 u 为根的子树中还在栈里的节点会构成一个点双，此时不断执行出栈操作直至点 u 出栈即可提取出这个点双中的点。不过，由于不同点双之间存在点的共用，点双的缩点在连边时不能像之前一样直接保留部分的边。具体来说：在缩点时先对每个点双连通分量建一个新点，然后让每两个有共用点的点双连通分量对应的新点连边。

点双缩点方法演示

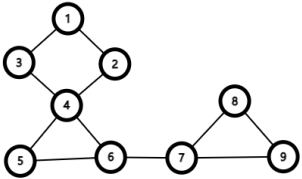


点双缩点方法演示



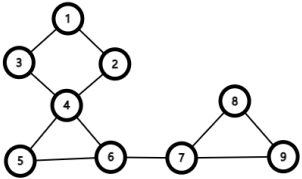
缩点后的图是 DAG 吗?

点双缩点方法演示



缩点后的图是 DAG 吗？不是（都不是有向的怎么 DAG）！

点双缩点方法演示



缩点后的图是 DAG 吗？不是（都不是有向的怎么 DAG）！具体来说，是一棵树。

「POJ 3177」Redundant Paths

给出一张 n 个点 m 条边的无向图，求至少在加入多少条边后可以变成边双连通图。
数据范围：$1 \le n \le 5\ 000, 1 \le m \le 10^4$。

「POJ 3177」Redundant Paths

给出一张 n 个点 m 条边的无向图，求至少在加入多少条边后可以变成边双连通图。

数据范围：$1 \le n \le 5\,000, 1 \le m \le 10^4$。

类比上一道例题，这题可以将所有的边双缩点形成一棵树。由于度数为 1 的点只能由别人到达它或者从它出发到达别人，肯定无法构成边双，而在这些度数为 1 的点两两连上边后，其路径上的点也会成为边双的一部分。所以问题转化为用最少的边使得图上不存在度数为 1 的点。

「POJ 3177」Redundant Paths

给出一张 n 个点 m 条边的无向图，求至少在加入多少条边后可以变成边双连通图。

数据范围：$1 \le n \le 5\,000, 1 \le m \le 10^4$。

类比上一道例题，这题可以将所有的边双缩点形成一棵树。由于度数为 1 的点只能由别人到达它或者从它出发到达别人，肯定无法构成边双，而在这些度数为 1 的点两两连上边后，其路径上的点也会成为边双的一部分。所以问题转化为用最少的边使得图上不存在度数为 1 的点。

此时最优策略为让度数为 1 的点两两配对。若有 x 个度数为 1 的点，那么答案就是 $\lceil \frac{x}{2} \rceil$。

「POJ 3177」Redundant Paths

给出一张 n 个点 m 条边的无向图，求至少在加入多少条边后可以变成边双连通图。

数据范围：$1 \le n \le 5\ 000, 1 \le m \le 10^4$。

类比上一道例题，这题可以将所有的边双缩点形成一棵树。由于度数为 1 的点只能由别人到达它或者从它出发到达别人，肯定无法构成边双，而在这些度数为 1 的点两两连上边后，其路径上的点也会成为边双的一部分。所以问题转化为用最少的边使得图上不存在度数为 1 的点。

此时最优策略为让度数为 1 的点两两配对。若有 x 个度数为 1 的点，那么答案就是 $\lceil \frac{x}{2} \rceil$。

注意：度数为 1 的点除了叶节点外，还有可能是只有一个子节点的根节点。

「HNOI 2012」矿场搭建

给出一张 n 个点 m 条边的无向图，求至少选择几个关键点使得在删除任意一个点后，每个点至少与一个关键点连通，并输出这种情况下有多少种不同的选择方案。

数据范围：$1 \leq n \leq 5 \times 10^6$。

「HNOI 2012」矿场搭建

给出一张 n 个点 m 条边的无向图，求至少选择几个关键点使得在删除任意一个点后，每个点至少与一个关键点连通，并输出这种情况下有多少种不同的选择方案。

数据范围：$1 \leq n \leq 5 \times 10^6$。

这题的核心是在研究删点后的连通性问题，于是可以联想到割点。如果通过割点对整张图切分，那么可以得到很多个点双连通分量。根据定义，如果删的点不是割点，那么对全局的连通性都不会产生影响（所以以下只考虑删割点的情况）。

「HNOI 2012」矿场搭建

给出一张 n 个点 m 条边的无向图，求至少选择几个关键点使得在删除任意一个点后，每个点至少与一个关键点连通，并输出这种情况下有多少种不同的选择方案。

数据范围：$1 \leq n \leq 5 \times 10^6$。

这题的核心是在研究删点后的连通性问题，于是可以联想到割点。如果通过割点对整张图切分，那么可以得到很多个点双连通分量。根据定义，如果删的点不是割点，那么对全局的连通性都不会产生影响（所以以下只考虑删割点的情况）。

由于这题在处理保证点双连通分量内部的点与点双连通分量内的关键点连通和保证一个点与其它点双连通分量中的点连通的方式不同，所以考虑分类讨论。

「HNOI 2012」矿场搭建

给出一张 n 个点 m 条边的无向图，求至少选择几个关键点使得在删除任意一个点后，每个点至少与一个关键点连通，并输出这种情况下有多少种不同的选择方案。

数据范围：$1 \leq n \leq 5 \times 10^6$。

这题的核心是在研究删点后的连通性问题，于是可以联想到割点。如果通过割点对整张图切分，那么可以得到很多个点双连通分量。根据定义，如果删的点不是割点，那么对全局的连通性都不会产生影响（所以以下只考虑删割点的情况）。

由于这题在处理保证点双连通分量内部的点与点双连通分量内的关键点连通和保证一个点与其它点双连通分量中的点连通的方式不同，所以考虑分类讨论。

1. 如果整张图构成一个点双，那么在这个点双中应该选择两个关键点，这样在删掉一个关键点后其它的点可以与另一个关键点连通。

「HNOI 2012」矿场搭建

给出一张 n 个点 m 条边的无向图，求至少选择几个关键点使得在删除任意一个点后，每个点至少与一个关键点连通，并输出这种情况下有多少种不同的选择方案。

数据范围：$1 \leq n \leq 5 \times 10^6$。

「HNOI 2012」矿场搭建

给出一张 n 个点 m 条边的无向图，求至少选择几个关键点使得在删除任意一个点后，每个点至少与一个关键点连通，并输出这种情况下有多少种不同的选择方案。

数据范围：$1 \le n \le 5 \times 10^6$。

2. 如果图上有多个点双，由于点双内部有着较好的连通性，可暂时不用考虑内部的情况，所以将图上的点双缩点变成一棵树，此时删掉割点等于在树上删边。

「HNOI 2012」矿场搭建

给出一张 n 个点 m 条边的无向图，求至少选择几个关键点使得在删除任意一个点后，每个点至少与一个关键点连通，并输出这种情况下有多少种不同的选择方案。

数据范围：$1 \leq n \leq 5 \times 10^6$。

2. 如果图上有多个点双，由于点双内部有着较好的连通性，可暂时不用考虑内部的情况，所以将图上的点双缩点变成一棵树，此时删掉割点等于在树上删边。

于是问题转化为，给定一棵树，要求选若干个关键点，使得在删去树的一条边的情况下，每个点都与至少一个关键点连通。

「HNOI 2012」矿场搭建

给出一张 n 个点 m 条边的无向图，求至少选择几个关键点使得在删除任意一个点后，每个点至少与一个关键点连通，并输出这种情况下有多少种不同的选择方案。

数据范围：$1 \leq n \leq 5 \times 10^6$。

2. 如果图上有多个点双，由于点双内部有着较好的连通性，可暂时不用考虑内部的情况，所以将图上的点双缩点变成一棵树，此时删掉割点等于在树上删边。

于是问题转化为，给定一棵树，要求选若干个关键点，使得在删去树的一条边的情况下，每个点都与至少一个关键点连通。

注意到对于度数为 1 的节点，只要将与它相连的那条边删去就会导致这个点的孤立，所以每个度数为 1 的节点必定是关键点。

「HNOI 2012」矿场搭建

给出一张 n 个点 m 条边的无向图，求至少选择几个关键点使得在删除任意一个点后，每个点至少与一个关键点连通，并输出这种情况下有多少种不同的选择方案。

数据范围：$1 \leq n \leq 5 \times 10^6$。

2. 如果图上有多个点双，由于点双内部有着较好的连通性，可暂时不用考虑内部的情况，所以将图上的点双缩点变成一棵树，此时删掉割点等于在树上删边。

于是问题转化为，给定一棵树，要求选若干个关键点，使得在删去树的一条边的情况下，每个点都与至少一个关键点连通。

注意到对于度数为 1 的节点，只要将与它相连的那条边删去就会导致这个点的孤立，所以每个度数为 1 的节点必定是关键点。

对于度数大于 1 的节点，由于树的“边缘”上的点度数一定为 1，所以删去任意一条边后，这个点一定还能与一些度数为 1 的点连通。故可以得出结论，设置的关键点数就是度数为 1 的点的个数。

「HNOI 2012」矿场搭建

给出一张 n 个点 m 条边的无向图，求至少选择几个关键点使得在删除任意一个点后，每个点至少与一个关键点连通，并输出这种情况下有多少种不同的选择方案。

数据范围：$1 \le n \le 5\times10^6$。

2. 如果图上有多个点双，由于点双内部有着较好的连通性，可暂时不用考虑内部的情况，所以将图上的点双缩点变成一棵树，此时删掉割点等于在树上删边。

于是问题转化为，给定一棵树，要求选若干个关键点，使得在删去树的一条边的情况下，每个点都与至少一个关键点连通。

注意到对于度数为 1 的节点，只要将与它相连的那条边删去就会导致这个点的孤立，所以每个度数为 1 的节点必定是关键点。

对于度数大于 1 的节点，由于树的“边缘”上的点度数一定为 1，所以删去任意一条边后，这个点一定还能与一些度数为 1 的点连通。故可以得出结论，设置的关键点数就是度数为 1 的点的个数。

最后特别注意，题目中**没有说明原图是连通的**，所以最后要再用加法原理计算总的关键点数，用乘法原理计算总的方案数。

章节目录

SAT 问题

在图论问题中，有这样一种模型：有 n 个变量（记为 $x_1 \sim x_n$），并且每个变量有 k 种取值 $1, 2, 3, \cdots k$；同时这些变量的取值之间还有一些关系，例如 $x_i = a$，那么 $x_j \neq b$ 等等。

SAT 问题

在图论问题中，有这样一种模型：有 n 个变量（记为 $x_1 \sim x_n$），并且每个变量有 k 种取值 $1, 2, 3, \cdots k$；同时这些变量的取值之间还有一些关系，例如 $x_i = a$，那么 $x_j \neq b$ 等等。

对于这一类问题，我们先对每个变量的每种取值建一个点，令点 (u, v) 表示 $x_u = v$ 的情况。对于每个变量有 k 种取值的情况，对应的问题被称为 k–SAT 问题。除了 2–SAT 问题，当 k 取其它的值时，不会存在比暴力枚举每个变量的取值更优的答案。

2-SAT 问题的基本处理思路

2-SAT 问题的求解基本分为两步：

2-SAT 问题的基本处理思路

2-SAT 问题的求解基本分为两步：

1. 对每个变量的两种状态建出点（以下假设这两种状态为 0 和 1）并按照题目限制中的逻辑推理的关系连边。

2-SAT 问题的基本处理思路

2-SAT 问题的求解基本分为两步：

1. 对每个变量的两种状态建出点（以下假设这两种状态为 0 和 1）并按照题目限制中的逻辑推理的关系连边。

举个例子：有 n 个人打算去玩，但其中有两个人 x, y 关系不好，不会同时参加活动。那么逻辑关系就是：若 x 参加了活动则 y 一定没参加、若 y 参加了活动则 x 一定没参加（注意，反过来就不成立了）。如果用 0 表示一个人不参加活动、1 表示参加，那么连边就是 $(x,1) \to (y,0), (y,1) \to (x,0)$。

2-SAT 问题的基本处理思路

2-SAT 问题的求解基本分为两步：

1. 对每个变量的两种状态建出点（以下假设这两种状态为 0 和 1）并按照题目限制中的逻辑推理的关系连边。

举个例子：有 n 个人打算去玩，但其中有两个人 x, y 关系不好，不会同时参加活动。那么逻辑关系就是：若 x 参加了活动则 y 一定没参加、若 y 参加了活动则 x 一定没参加（注意，反过来就不成立了）。如果用 0 表示一个人不参加活动、1 表示参加，那么连边就是 $(x,1) \to (y,0), (y,1) \to (x,0)$。

2. 逐一枚举每一个不确定的变量，先假设这个变量取 0 并进行遍历，确定其他的变量的值。如果出现冲突（某个变量同时选了 0 和 1），那么就将这个操作撤回，令这个变量取 1。如果所有操作结束后，仍不能找到答案，那么说明这个情形无解。

算法优化

在步骤 2 中，如果对整张图直接进行枚举和遍历，则无法保证算法的复杂度。注意到出现问题的情况是**同时**能推出“若 x 去则 x 不去”和“若 x 不去则 x 去”的矛盾结论，那么就意味着此时 $(x,0)$ 与 $(x,1)$ 位于同一个强连通分量中。所以我们可以通过 Tarjan 对整张图进行一个初步的判断，确定是否可能存在解。

算法优化

在步骤 2 中，如果对整张图直接进行枚举和遍历，则无法保证算法的复杂度。注意到出现问题的情况是**同时**能推出“若 x 去则 x 不去”和“若 x 不去则 x 去”的矛盾结论，那么就意味着此时 $(x,0)$ 与 $(x,1)$ 位于同一个强连通分量中。所以我们可以通过 Tarjan 对整张图进行一个初步的判断，确定是否可能存在解。

接下来，由于强连通分量内的点一定是同时选取的，所以可以先将整张图缩点再进行进一步处理。

算法优化

在步骤 2 中，如果对整张图直接进行枚举和遍历，则无法保证算法的复杂度。注意到出现问题的情况是**同时**能推出“若 x 去则 x 不去”和“若 x 不去则 x 去”的矛盾结论，那么就意味着此时 $(x,0)$ 与 $(x,1)$ 位于同一个强连通分量中。所以我们可以通过 Tarjan 对整张图进行一个初步的判断，确定是否可能存在解。

接下来，由于强连通分量内的点一定是同时选取的，所以可以先将整张图缩点再进行进一步处理。

若要求给出具体方案，则每次选择 $(x,0)$ 与 $(x,1)$ 拓扑序较大的一个点，就一定存在可行解（拓扑序小的点有较多的后继，如果选取拓扑序大的方案无解，那么选取拓扑序小的方案一定也无解）。

小测验 2

1. 若某个变量 x 一定为 1，那么在连边时要怎么体现?

小测验 2

1. 若某个变量 x 一定为 1，那么在连边时要怎么体现？
从 $(x,0)$ 向 $(x,1)$ 连边，因为这样一旦选 0 就会冲突而选 1 则不会。

小测验 2

1. 若某个变量 x 一定为 1，那么在连边时要怎么体现？
从 $(x,0)$ 向 $(x,1)$ 连边，因为这样一旦选 0 就会冲突而选 1 则不会。
2. 如何判断某个局面是否存在唯一解？

小测验 2

1. 若某个变量 x 一定为 1，那么在连边时要怎么体现？

从 $(x,0)$ 向 $(x,1)$ 连边，因为这样一旦选 0 就会冲突而选 1 则不会。

2. 如何判断某个局面是否存在唯一解？

先跑出一组可行解，然后枚举每一个变量反选的情况，如果存在另外一组可行解，那么存在多解。

「CF776D」The Door Problem

有 n 扇门和 m 个开关，门的状态只有开和关两种，初始时只有部分的门是开的。每个开关可以控制多扇门，按下后可以改变对应的门的开关状态（开的关闭，关的开启）。保证每扇门只受两个开关控制，求最少多少次操作后所有的门都会被打开。

数据范围：$n, m \leq 10^5$。

「CF776D」The Door Problem

有 n 扇门和 m 个开关，门的状态只有开和关两种，初始时只有部分的门是开的。每个开关可以控制多扇门，按下后可以改变对应的门的开关状态（开的关闭，关的开启）。保证每扇门只受两个开关控制，求最少多少次操作后所有的门都会被打开。

数据范围：$n, m \leq 10^5$。

每扇门只有两个状态（开和关），每个开关也只有两个状态（操作和不操作，因为操作偶数次等于不操作，奇数次等于操作一次）。两种变量看起来都适合做 2-SAT。

「CF776D」The Door Problem

有 n 扇门和 m 个开关，门的状态只有开和关两种，初始时只有部分的门是开的。每个开关可以控制多扇门，按下后可以改变对应的门的开关状态（开的关闭，关的开启）。保证每扇门只受两个开关控制，求最少多少次操作后所有的门都会被打开。

数据范围：$n, m \leq 10^5$。

每扇门只有两个状态（开和关），每个开关也只有两个状态（操作和不操作，因为操作偶数次等于不操作，奇数次等于操作一次）。两种变量看起来都适合做 2-SAT。

接下来考虑如何用条件在不同的状态间连边。注意到题目中的特殊条件“保证每扇门只受两个开关控制”，这说明如果想要门的状态不变，那么两个开关一定都按了或都没按；如果想要改变门的状态，那么两个开关肯定一个有按一个不按。

「CF776D」The Door Problem

有 n 扇门和 m 个开关，门的状态只有开和关两种，初始时只有部分的门是开的。每个开关可以控制多扇门，按下后可以改变对应的门的开关状态（开的关闭，关的开启）。保证每扇门只受两个开关控制，求最少多少次操作后所有的门都会被打开。

数据范围：$n, m \leq 10^5$。

每扇门只有两个状态（开和关），每个开关也只有两个状态（操作和不操作，因为操作偶数次等于不操作，奇数次等于操作一次）。两种变量看起来都适合做 2-SAT。

接下来考虑如何用条件在不同的状态间连边。注意到题目中的特殊条件“保证每扇门只受两个开关控制”，这说明如果想要门的状态不变，那么两个开关一定都按了或都没按；如果想要改变门的状态，那么两个开关肯定一个有按一个不按。

具体地，我们用 0 和 1 表示一个开关没按/按了。假设某一扇门由开关 x, y 控制。如果这扇门一开始是关的，那么 x 与 y 一定一个取 0 一个取 1，即连边为 $(x, 0) \leftrightarrow (y, 1), (x, 1) \leftrightarrow (y, 0)$；如果这扇门开始的时候已经开了，那么 x 与 y 一定取值相等，连边为 $(x, 0) \leftrightarrow (y, 0), (x, 1) \leftrightarrow (y, 1)$。之后跑 2-SAT 求解即可。

「NOI 2017」游戏

有 n 张地图，每张地图有一个属性 $t \in \{a, b, c, x\}$，同时，你还有 3 辆车，每一辆车也有一种属性 $t \in \{A, B, C\}$，已知 A 车不能在 a 图上跑，B 车不能在 b 图上跑，C 车不能在 c 图上跑，x 图适合任意一种车。此外还有 m 个限制 (i, h_i, j, h_j) 表示如果地图 i 用了车 h_i，那么地图 j 一定要用车 h_j。请输出任意一种安排方案。

数据范围：$1 \le n \le 5 \times 10^4, 1 \le m \le 10^5$，记 $d = \sum[t == x] \le 8$。

「NOI 2017」游戏

有 n 张地图，每张地图有一个属性 $t \in \{a, b, c, x\}$，同时，你还有 3 辆车，每一辆车也有一种属性 $t \in \{A, B, C\}$，已知 A 车不能在 a 图上跑，B 车不能在 b 图上跑，C 车不能在 c 图上跑，x 图适合任意一种车。此外还有 m 个限制 (i, h_i, j, h_j) 表示如果地图 i 用了车 h_i，那么地图 j 一定要用车 h_j。请输出任意一种安排方案。

数据范围：$1 \le n \le 5 \times 10^4, 1 \le m \le 10^5$，记 $d = \sum [t == x] \le 8$。

这是一个 3-SAT 问题，理论上是只能暴力枚举的，肯定无法处理 n 和 m 如此大的情况。

「NOI 2017」游戏

有 n 张地图，每张地图有一个属性 $t \in \{a, b, c, x\}$，同时，你还有 3 辆车，每一辆车也有一种属性 $t \in \{A, B, C\}$，已知 A 车不能在 a 图上跑，B 车不能在 b 图上跑，C 车不能在 c 图上跑，x 图适合任意一种车。此外还有 m 个限制 (i, h_i, j, h_j) 表示如果地图 i 用了车 h_i，那么地图 j 一定要用车 h_j。请输出任意一种安排方案。

数据范围：$1 \le n \le 5 \times 10^4, 1 \le m \le 10^5$，记 $d = \sum[t == x] \le 8$。

这是一个 3-SAT 问题，理论上是只能暴力枚举的，肯定无法处理 n 和 m 如此大的情况。

如果在题目中找不到思路，可以从数据范围找突破口，如用复杂度要求猜测可能要用算法或寻找特殊的数据范围。

「NOI 2017」游戏

有 n 张地图，每张地图有一个属性 $t \in \{a, b, c, x\}$，同时，你还有 3 辆车，每一辆车也有一种属性 $t \in \{A, B, C\}$，已知 A 车不能在 a 图上跑，B 车不能在 b 图上跑，C 车不能在 c 图上跑，x 图适合任意一种车。此外还有 m 个限制 (i, h_i, j, h_j) 表示如果地图 i 用了车 h_i，那么地图 j 一定要用车 h_j。请输出任意一种安排方案。

数据范围：$1 \le n \le 5 \times 10^4, 1 \le m \le 10^5$，记 $d = \sum[t == x] \le 8$。

这是一个 3-SAT 问题，理论上是只能暴力枚举的，肯定无法处理 n 和 m 如此大的情况。

如果在题目中找不到思路，可以从数据范围找突破口，如用复杂度要求猜测可能要用算法或寻找特殊的数据范围。

在这题的数据范围中，最特殊一点是 $\sum[t == x] \le 8$，即 x 图不超过 8 张，这是一个看上去对暴力很友好的数据范围。于是可以直接枚举这些 x 图的选车情况，剩下的 a, b, c 图按正常的 2-SAT 做即可。时间复杂度 $\mathbb{O}(3^d(m+n))$。

「NOI 2017」游戏

有 n 张地图，每张地图有一个属性 $t \in \{a, b, c, x\}$，同时，你还有 3 辆车，每一辆车也有一种属性 $t \in \{A, B, C\}$，已知 A 车不能在 a 图上跑，B 车不能在 b 图上跑，C 车不能在 c 图上跑，x 图适合任意一种车。此外还有 m 个限制 (i, h_i, j, h_j) 表示如果地图 i 用了车 h_i，那么地图 j 一定要用车 h_j。请输出任意一种安排方案。

数据范围：$1 \le n \le 5 \times 10^4, 1 \le m \le 10^5, \sum[t == x] \le 8$。

「NOI 2017」游戏

有 n 张地图，每张地图有一个属性 $t \in \{a, b, c, x\}$，同时，你还有 3 辆车，每一辆车也有一种属性 $t \in \{A, B, C\}$，已知 A 车不能在 a 图上跑，B 车不能在 b 图上跑，C 车不能在 c 图上跑，x 图适合任意一种车。此外还有 m 个限制 (i, h_i, j, h_j) 表示如果地图 i 用了车 h_i，那么地图 j 一定要用车 h_j。请输出任意一种安排方案。

数据范围：$1 \le n \le 5 \times 10^4, 1 \le m \le 10^5, \sum[t == x] \le 8$。

但是，我们真的要枚举所有的情况吗?

「NOI 2017」游戏

有 n 张地图，每张地图有一个属性 $t \in \{a, b, c, x\}$，同时，你还有 3 辆车，每一辆车也有一种属性 $t \in \{A, B, C\}$，已知 A 车不能在 a 图上跑，B 车不能在 b 图上跑，C 车不能在 c 图上跑，x 图适合任意一种车。此外还有 m 个限制 (i, h_i, j, h_j) 表示如果地图 i 用了车 h_i，那么地图 j 一定要用车 h_j。请输出任意一种安排方案。

数据范围：$1 \le n \le 5 \times 10^4, 1 \le m \le 10^5, \sum[t == x] \le 8$。

但是，我们真的要枚举所有的情况吗？注意到 2-SAT 问题可以处理每个变量有两种取值的情况，所以我们不一定要枚举出 x 图的具体选车情况，而可以留两种选择给 2-SAT 解决。具体来说，我们可以先把某一个 x 图当成 a 图跑 2-SAT，如果最后发现无论其它的地图怎么选择都是无解的，那么这张 x 图上跑的一定是 A 车。这样优化后的枚举方式可以让每张 x 图的可能性降到两种（“是 a 图” 和 “跑 A 车”），时间复杂度降低到 $\mathbb{O}(2^d(m+n))$。

「NOI 2017」游戏 · 技术总结

1. 对题目没有思路的时候可以从数据范围找突破口。

「NOI 2017」游戏 · 技术总结

1. 对题目没有思路的时候可以从数据范围找突破口。
2. 对于已知无解的问题（如 3-SAT），为了让题目可做，其中肯定含有一些特殊性质，这一般也是题目的突破口。

「NOI 2017」游戏 · 技术总结

1. 对题目没有思路的时候可以从数据范围找突破口。
2. 对于已知无解的问题（如 3-SAT），为了让题目可做，其中肯定含有一些特殊性质，这一般也是题目的突破口。
3. 对于不会的问题（如 3-SAT），可以向相似且可做的问题的方向思考（如 2-SAT）。

完结撒花