图论中的连通性问题

华师大二附中 管晏如

2023 年 7 月 18 日

目录

引入

- 并查集是一种维护集合的数据结构，支持的操作有：

引入

- 并查集是一种维护集合的数据结构，支持的操作有：
  - 合并两个点所在的集合；

引入

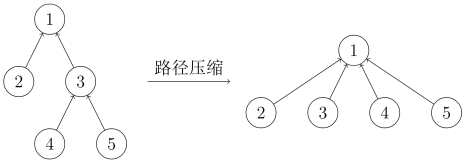
- 并查集是一种维护集合的数据结构，支持的操作有：
  - 合并两个点所在的集合；
  - 查询两个点是否在同一个集合。

引入

- 并查集是一种维护集合的数据结构，支持的操作有：
  - 合并两个点所在的集合；
  - 查询两个点是否在同一个集合。
- 常规的实现方法：

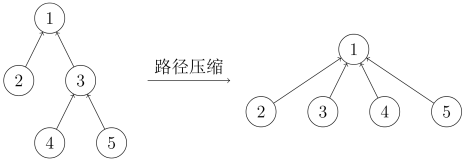
引入

- 并查集是一种维护集合的数据结构，支持的操作有：
  - 合并两个点所在的集合；
  - 查询两个点是否在同一个集合。
- 常规的实现方法：
  - 路径压缩；



引入

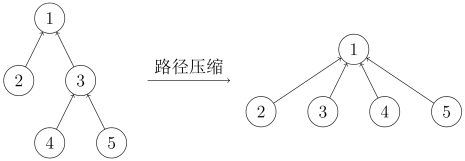
- 并查集是一种维护集合的数据结构，支持的操作有：
  - 合并两个点所在的集合；
  - 查询两个点是否在同一个集合。
- 常规的实现方法：
  - 路径压缩；



  - 启发式合并。

引入

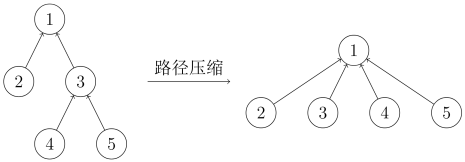
- 并查集是一种维护集合的数据结构，支持的操作有：
  - 合并两个点所在的集合；
  - 查询两个点是否在同一个集合。
- 常规的实现方法：
  - 路径压缩；



  - 启发式合并。
- 时间复杂度：以上两种使用一种为 $O(n \log n)$，同时使用为 $O(n\alpha(n))$。

引入

- 并查集是一种维护集合的数据结构，支持的操作有：
  - 合并两个点所在的集合；
  - 查询两个点是否在同一个集合。
- 常规的实现方法：
  - 路径压缩；



  - 启发式合并。
- 时间复杂度：以上两种使用一种为 $O(n \log n)$，同时使用为 $O(n\alpha(n))$。
- 可同时维护点权（或边权），称为带权并查集。

例 1

题目大意

- 有一个长度为 n 的 01 序列 $\{a_n\}$，初始全为 0。
- 下面进行 m 次操作，每次将一个 $a[x]$ 修改为 1，并查询 $a[x], a[x+1], ..., a[n]$ 中第一个 0 的位置。

例 1

题目大意

- 有一个长度为 n 的 01 序列 $\{a_n\}$，初始全为 0。
- 下面进行 m 次操作，每次将一个 $a[x]$ 修改为 1，并查询 $a[x], a[x+1], ..., a[n]$ 中第一个 0 的位置。

- 法 1：使用分块。

例 1

题目大意

- 有一个长度为 n 的 01 序列 $\{a_n\}$，初始全为 0。
- 下面进行 m 次操作，每次将一个 $a[x]$ 修改为 1，并查询 $a[x], a[x+1], ..., a[n]$ 中第一个 0 的位置。

- 法 1：使用分块。
- 法 2：使用 set。

例 1

题目大意

- 有一个长度为 n 的 01 序列 $\{a_n\}$，初始全为 0。
- 下面进行 m 次操作，每次将一个 $a[x]$ 修改为 1，并查询 $a[x], a[x+1], ..., a[n]$ 中第一个 0 的位置。

- 法 1：使用分块。
- 法 2：使用 set。
- 法 3：使用并查集维护 f_i 表示 $a[i], a[i+1], ..., a[n]$ 中第一个 0 的位置。

例 1

题目大意

- 有一个长度为 n 的 01 序列 $\{a_n\}$，初始全为 0。
- 下面进行 m 次操作，每次将一个 $a[x]$ 修改为 1，并查询 $a[x], a[x+1], ..., a[n]$ 中第一个 0 的位置。

- 法 1：使用分块。
- 法 2：使用 set。
- 法 3：使用并查集维护 f_i 表示 $a[i], a[i+1], ..., a[n]$ 中第一个 0 的位置。
- 初始 $f[i] = i$。每次操作即为令 $f[x] = f[x+1]$，然后查询 $f[x]$。

例 2

题目大意

- 给定三个长度为 n 的整数序列 $\{a_n\}, \{b_n\}, \{c_n\}$。
- 对于 $1 \leq i \leq j \leq n$，求 $a_i b_j \max_{i \leq k \leq j} c_k$ 的最大值。

例 2

题目大意

- 给定三个长度为 n 的整数序列 $\{a_n\}, \{b_n\}, \{c_n\}$。
- 对于 $1 \leq i \leq j \leq n$，求 $a_i b_j \max_{i \leq k \leq j} c_k$ 的最大值。

- 枚举区间最大值 c_k 的位置 k。

例 2

题目大意

- 给定三个长度为 n 的整数序列 $\{a_n\}, \{b_n\}, \{c_n\}$。
- 对于 $1 \le i \le j \le n$，求 $a_i b_j \max_{i \le k \le j} c_k$ 的最大值。

- 枚举区间最大值 c_k 的位置 k。
- 法 1：使用单调栈，找到最左和最右第一个 $> c_k$ 的，随后相当于求区间的最大、最小值，可以使用 ST 表预处理。

例 2

题目大意

- 给定三个长度为 n 的整数序列 $\{a_n\}, \{b_n\}, \{c_n\}$。
- 对于 $1 \le i \le j \le n$，求 $a_i b_j \max_{i \le k \le j} c_k$ 的最大值。

- 枚举区间最大值 c_k 的位置 k。
- 法 1：使用单调栈，找到最左和最右第一个 $> c_k$ 的，随后相当于求区间的最大、最小值，可以使用 ST 表预处理。
- 法 2：按 c_k 从小到大的顺序枚举 k，每次相当于激活一个位置，然后查询当前激活的极长连续段中最大、最小值是多少。

例 2

题目大意

- 给定三个长度为 n 的整数序列 $\{a_n\}, \{b_n\}, \{c_n\}$。
- 对于 $1 \le i \le j \le n$，求 $a_i b_j \max_{i \le k \le j} c_k$ 的最大值。

- 枚举区间最大值 c_k 的位置 k。
- 法 1：使用单调栈，找到最左和最右第一个 $> c_k$ 的，随后相当于求区间的最大、最小值，可以使用 ST 表预处理。
- 法 2：按 c_k 从小到大的顺序枚举 k，每次相当于激活一个位置，然后查询当前激活的极长连续段中最大、最小值是多少。可以使用并查集维护连续段，并可以在合并的同时维护最大、最小值。

CF 1444 C

题目大意

给定一张 n 个点 m 条边的无向图，无重边自环。每个点被染上了一种颜色，共 k 种不同的颜色。
求有多少对不同颜色的 pair，使得取出这两种颜色的所有点，构成的子图为二分图。
范围：$n, m, k \leq 5 \times 10^5$

CF 1444 C

题目大意

给定一张 n 个点 m 条边的无向图，无重边自环。每个点被染上了一种颜色，共 k 种不同的颜色。
求有多少对不同颜色的 pair，使得取出这两种颜色的所有点，构成的子图为二分图。
范围：$n, m, k \leq 5 \times 10^5$

- 首先，对于每种颜色，检查其自身构成的子图是否为二分图。不是的话显然可以直接删除该种颜色。

CF 1444 C

题目大意

给定一张 n 个点 m 条边的无向图，无重边自环。每个点被染上了一种颜色，共 k 种不同的颜色。
求有多少对不同颜色的 pair，使得取出这两种颜色的所有点，构成的子图为二分图。
范围：$n, m, k \leq 5 \times 10^5$

- 首先，对于每种颜色，检查其自身构成的子图是否为二分图。不是的话显然可以直接删除该种颜色。
- 此时如果两种颜色之间根本没边，一定合法。

CF 1444 C

题目大意

给定一张 n 个点 m 条边的无向图，无重边自环。每个点被染上了一种颜色，共 k 种不同的颜色。
求有多少对不同颜色的 pair，使得取出这两种颜色的所有点，构成的子图为二分图。
范围：$n, m, k \leq 5 \times 10^5$

- 首先，对于每种颜色，检查其自身构成的子图是否为二分图。不是的话显然可以直接删除该种颜色。
- 此时如果两种颜色之间根本没边，一定合法。反过来，只有存在一条边 $(x, y), c[x] \neq c[y]$，颜色对 $(c[x], c[y])$ 才有可能不合法。

CF 1444 C

题目大意

给定一张 n 个点 m 条边的无向图，无重边自环。每个点被染上了一种颜色，共 k 种不同的颜色。
求有多少对不同颜色的 pair，使得取出这两种颜色的所有点，构成的子图为二分图。
范围：$n, m, k \leq 5 \times 10^5$

- 首先，对于每种颜色，检查其自身构成的子图是否为二分图。不是的话显然可以直接删除该种颜色。
- 此时如果两种颜色之间根本没边，一定合法。反过来，只有存在一条边 $(x, y), c[x] \neq c[y]$，颜色对 $(c[x], c[y])$ 才有可能不合法。这样的 pair 不超过 $O(m)$ 个，可以枚举，然后判断是否是二分图。

CF 1444 C

题目大意

给定一张 n 个点 m 条边的无向图，无重边自环。每个点被染上了一种颜色，共 k 种不同的颜色。
求有多少对不同颜色的 pair，使得取出这两种颜色的所有点，构成的子图为二分图。
范围：$n, m, k \leq 5 \times 10^5$

- 首先，对于每种颜色，检查其自身构成的子图是否为二分图。不是的话显然可以直接删除该种颜色。
- 此时如果两种颜色之间根本没边，一定合法。反过来，只有存在一条边 $(x, y), c[x] \neq c[y]$，颜色对 $(c[x], c[y])$ 才有可能不合法。这样的 pair 不超过 $O(m)$ 个，可以枚举，然后判断是否是二分图。
- 具体来说，对于任何一条边 (x, y)，我们在并查集中连接 $(x, y+n)$ 与 $(x+n, y)$。如果在加入 (x, y) 之前 x, y 已经在并查集内连通，则说明存在奇环。

CF 1444 C

题目大意

给定一张 n 个点 m 条边的无向图，无重边自环。每个点被染上了一种颜色，共 k 种不同的颜色。
求有多少对不同颜色的 pair，使得取出这两种颜色的所有点，构成的子图为二分图。
范围：$n, m, k \leq 5 \times 10^5$

- 首先，对于每种颜色，检查其自身构成的子图是否为二分图。不是的话显然可以直接删除该种颜色。
- 此时如果两种颜色之间根本没边，一定合法。反过来，只有存在一条边 $(x, y), c[x] \neq c[y]$，颜色对 $(c[x], c[y])$ 才有可能不合法。这样的 pair 不超过 $O(m)$ 个，可以枚举，然后判断是否是二分图。
- 具体来说，对于任何一条边 (x, y)，我们在并查集中连接 $(x, y+n)$ 与 $(x+n, y)$。如果在加入 (x, y) 之前 x, y 已经在并查集内连通，则说明存在奇环。
- 因此，我们先对于并查集加入所有 $c[x] = c[y]$ 的边。然后枚举可能不合法的颜色 pair，同时加入这两种颜色之间的边，等 check 完了再撤销。

CF 1444 C

题目大意

给定一张 n 个点 m 条边的无向图，无重边自环。每个点被染上了一种颜色，共 k 种不同的颜色。
求有多少对不同颜色的 pair，使得取出这两种颜色的所有点，构成的子图为二分图。
范围：$n, m, k \leq 5 \times 10^5$

- 首先，对于每种颜色，检查其自身构成的子图是否为二分图。不是的话显然可以直接删除该种颜色。
- 此时如果两种颜色之间根本没边，一定合法。反过来，只有存在一条边 $(x, y), c[x] \neq c[y]$，颜色对 $(c[x], c[y])$ 才有可能不合法。这样的 pair 不超过 $O(m)$ 个，可以枚举，然后判断是否是二分图。
- 具体来说，对于任何一条边 (x, y)，我们在并查集中连接 $(x, y+n)$ 与 $(x+n, y)$。如果在加入 (x, y) 之前 x, y 已经在并查集内连通，则说明存在奇环。
- 因此，我们先对于并查集加入所有 $c[x] = c[y]$ 的边。然后枚举可能不合法的颜色 pair，同时加入这两种颜色之间的边，等 check 完了再撤销。
- 并查集要支持撤销，应当是启发式合并的。

CODE FESTIVAL 2017 Final E

题目大意

给定一个字符串 S，由 26 个小写字母组成。
允许进行 n 种操作（以任何顺序、任何次数），每种操作用 l_i, r_i 表示：将 $S[l_i], S[l_i+1], ..., S[r_i]$ 全部 $+1$，若是字母 z 则变为字母 a。
问是否可以将 S 变为回文串。
范围：$1 \leq |S|, n \leq 10^5$。

CODE FESTIVAL 2017 Final E

题目大意

给定一个字符串 S，由 26 个小写字母组成。
允许进行 n 种操作（以任何顺序、任何次数），每种操作用 l_i, r_i 表示：将 $S[l_i], S[l_i+1], ..., S[r_i]$ 全部 $+1$，若是字母 z 则变为字母 a。
问是否可以将 S 变为回文串。
范围：$1 \leq |S|, n \leq 10^5$。

- 差分。

CODE FESTIVAL 2017 Final E

题目大意

给定一个字符串 S，由 26 个小写字母组成。
允许进行 n 种操作（以任何顺序、任何次数），每种操作用 l_i, r_i 表示：将 $S[l_i], S[l_i+1], ..., S[r_i]$ 全部 $+1$，若是字母 z 则变为字母 a。
问是否可以将 S 变为回文串。
范围：$1 \leq |S|, n \leq 10^5$。

- 差分。区间加法相当于前面 $+1$，后面 -1。回文相当于对称的位置恰为相反数。

CODE FESTIVAL 2017 Final E

题目大意

给定一个字符串 S，由 26 个小写字母组成。
允许进行 n 种操作（以任何顺序、任何次数），每种操作用 l_i, r_i 表示：将 $S[l_i], S[l_i+1], ..., S[r_i]$ 全部 $+1$，若是字母 z 则变为字母 a。
问是否可以将 S 变为回文串。
范围：$1 \leq |S|, n \leq 10^5$。

- 差分。区间加法相当于前面 $+1$，后面 -1。回文相当于对称的位置恰为相反数。
- 那么我们把需要是相反数的位置连起来，然后把 l 和 $r+1$ 也都连起来，则有解当且仅当每个连通块的初值和 $= 0$。

洛谷 3295 [SCOI2016] 萌萌哒

题目大意

有一个长度为 n 的字符串 S，每一位可以取 $0, 1, ..., 9$。
现给出 m 个限制条件，形如 $S[l_1...r_1] = S[l_2...r_2]$，问 S 有多少种不同的可能。答案对 $10^9 + 7$ 取模。
范围：$1 \leq n, m \leq 10^5$。

洛谷 3295 [SCOI2016] 萌萌哒

题目大意

有一个长度为 n 的字符串 S，每一位可以取 $0, 1, ..., 9$。
现给出 m 个限制条件，形如 $S[l_1...r_1] = S[l_2...r_2]$，问 S 有多少种不同的可能。答案对 $10^9 + 7$ 取模。
范围：$1 \leq n, m \leq 10^5$。

- 我们需要快速求出并查集连接后的结果。

洛谷 3295 [SCOI2016] 萌萌哒

题目大意

有一个长度为 n 的字符串 S，每一位可以取 $0, 1, ..., 9$。
现给出 m 个限制条件，形如 $S[l_1...r_1] = S[l_2...r_2]$，问 S 有多少种不同的可能。答案对 $10^9 + 7$ 取模。
范围：$1 \leq n, m \leq 10^5$。

- 我们需要快速求出并查集连接后的结果。
- 考虑 ST 表的思想。令每个位置 i 开始、长度为 2^k 的区间为一个点，那么每条限制根据长度作二进制分解，就可以变成 $\log n$ 条新图中的边。

洛谷 3295 [SCOI2016] 萌萌哒

题目大意

有一个长度为 n 的字符串 S，每一位可以取 $0, 1, ..., 9$。
现给出 m 个限制条件，形如 $S[l_1...r_1] = S[l_2...r_2]$，问 S 有多少种不同的可能。答案对 $10^9 + 7$ 取模。
范围：$1 \leq n, m \leq 10^5$。

- 我们需要快速求出并查集连接后的结果。
- 考虑 ST 表的思想。令每个位置 i 开始、长度为 2^k 的区间为一个点，那么每条限制根据长度作二进制分解，就可以变成 $\log n$ 条新图中的边。
- 最后，我们在新图中进行限制关系的下放

洛谷 3295 [SCOI2016] 萌萌哒

题目大意

有一个长度为 n 的字符串 S，每一位可以取 $0, 1, ..., 9$。
现给出 m 个限制条件，形如 $S[l_1...r_1] = S[l_2...r_2]$，问 S 有多少种不同的可能。答案对 $10^9 + 7$ 取模。
范围：$1 \leq n, m \leq 10^5$。

- 我们需要快速求出并查集连接后的结果。
- 考虑 ST 表的思想。令每个位置 i 开始、长度为 2^k 的区间为一个点，那么每条限制根据长度作二进制分解，就可以变成 $\log n$ 条新图中的边。
- 最后，我们在新图中进行限制关系的下放，注意到我们可以始终保持连边是同层（k 相等）之间的

洛谷 3295 [SCOI2016] 萌萌哒

题目大意

有一个长度为 n 的字符串 S，每一位可以取 $0, 1, ..., 9$。
现给出 m 个限制条件，形如 $S[l_1...r_1] = S[l_2...r_2]$，问 S 有多少种不同的可能。答案对 $10^9 + 7$ 取模。
范围：$1 \leq n, m \leq 10^5$。

- 我们需要快速求出并查集连接后的结果。
- 考虑 ST 表的思想。令每个位置 i 开始、长度为 2^k 的区间为一个点，那么每条限制根据长度作二进制分解，就可以变成 $\log n$ 条新图中的边。
- 最后，我们在新图中进行限制关系的下放，注意到我们可以始终保持连边是同层（k 相等）之间的，每次只要把 $(i, 2^k)$ 与 $(find(i), 2^k)$ 的相等关系，下放到左右两个 2^{k-1} 区间之间就可以了。

并查集进阶

- 使用主席树维护可持久化的并查集；
- Kruskal 重构树；
- 线性时间复杂度维护序列/树上并查集；
- ...
- 感兴趣的同学可以自行学习。

引入

引入

- 在一个有向图中，如果有一些点之间两两可达，则称它们强联通。

引入

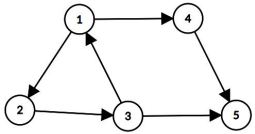
- 在一个有向图中，如果有一些点之间两两可达，则称它们强联通。
- 强联通分量（SCC）就是极大的强联通子图。

引入

- 在一个有向图中，如果有一些点之间两两可达，则称它们强联通。
- 强联通分量（SCC）就是极大的强联通子图。
- 将强联通分量缩起来，可以得到一张 DAG。

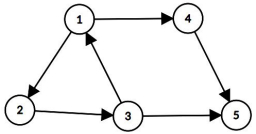
引入

- 在一个有向图中，如果有一些点之间两两可达，则称它们强联通。
- 强联通分量（SCC）就是极大的强联通子图。
- 将强联通分量缩起来，可以得到一张 DAG。



引入

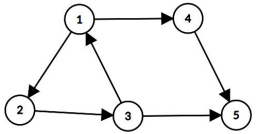
- 在一个有向图中，如果有一些点之间两两可达，则称它们强联通。
- 强联通分量（SCC）就是极大的强联通子图。
- 将强联通分量缩起来，可以得到一张 DAG。



- 下面介绍一种求 SCC 的方法：Tarjan 算法，其适用性是最广泛的。

引入

- 在一个有向图中，如果有一些点之间两两可达，则称它们强联通。
- 强联通分量（SCC）就是极大的强联通子图。
- 将强联通分量缩起来，可以得到一张 DAG。



- 下面介绍一种求 SCC 的方法：Tarjan 算法，其适用性是最广泛的。
- 其他例如 Kosaraju 算法同学们可以课下自行学习。

Tarjan 算法

Tarjan 算法

- 在该算法中，我们为每个节点 u 维护以下变量：

Tarjan 算法

- 在该算法中，我们为每个节点 u 维护以下变量：
  - $dfn[u]$ 表示 dfs 时 u 是第几个被遍历到的；

Tarjan 算法

- 在该算法中，我们为每个节点 u 维护以下变量：
  - $dfn[u]$ 表示 dfs 时 u 是第几个被遍历到的；
  - $low[u]$ 表示 u 子树内跳到子树外的 dfn 最小值，初值等于 $dfn[u]$。

Tarjan 算法

- 在该算法中，我们为每个节点 u 维护以下变量：
  - $dfn[u]$ 表示 dfs 时 u 是第几个被遍历到的；
  - $low[u]$ 表示 u 子树内跳到子树外的 dfn 最小值，初值等于 $dfn[u]$。
- 易见，一个子树内的 dfn 值都严格大于子树根。

Tarjan 算法

- 在该算法中，我们为每个节点 u 维护以下变量：
  - $dfn[u]$ 表示 dfs 时 u 是第几个被遍历到的；
  - $low[u]$ 表示 u 子树内跳到子树外的 dfn 最小值，初值等于 $dfn[u]$。
- 易见，一个子树内的 dfn 值都严格大于子树根。
- 在 dfs 的过程中，除了维护 dfn 和 low，我们需要将搜索到的点都放入一个栈内。随后，在每发现一个 SCC 时，就不断弹栈，直到这个 SCC 被完整弹出。

Tarjan 算法

- 在该算法中，我们为每个节点 u 维护以下变量：
  - $dfn[u]$ 表示 dfs 时 u 是第几个被遍历到的；
  - $low[u]$ 表示 u 子树内跳到子树外的 dfn 最小值，初值等于 $dfn[u]$。
- 易见，一个子树内的 dfn 值都严格大于子树根。
- 在 dfs 的过程中，除了维护 dfn 和 low，我们需要将搜索到的点都放入一个栈内。随后，在每发现一个 SCC 时，就不断弹栈，直到这个 SCC 被完整弹出。
- 从 u 搜索到其儿子 v 时，考虑以下几种情况：

Tarjan 算法

- 在该算法中，我们为每个节点 u 维护以下变量：
  - $dfn[u]$ 表示 dfs 时 u 是第几个被遍历到的；
  - $low[u]$ 表示 u 子树内跳到子树外的 dfn 最小值，初值等于 $dfn[u]$。
- 易见，一个子树内的 dfn 值都严格大于子树根。
- 在 dfs 的过程中，除了维护 dfn 和 low，我们需要将搜索到的点都放入一个栈内。随后，在每发现一个 SCC 时，就不断弹栈，直到这个 SCC 被完整弹出。
- 从 u 搜索到其儿子 v 时，考虑以下几种情况：
  - v 没被搜索过：直接进入 v 进行 dfs，然后令 $low[u] = \min(low[u], low[v])$；

Tarjan 算法

- 在该算法中，我们为每个节点 u 维护以下变量：
  - $dfn[u]$ 表示 dfs 时 u 是第几个被遍历到的；
  - $low[u]$ 表示 u 子树内跳到子树外的 dfn 最小值，初值等于 $dfn[u]$。
- 易见，一个子树内的 dfn 值都严格大于子树根。
- 在 dfs 的过程中，除了维护 dfn 和 low，我们需要将搜索到的点都放入一个栈内。随后，在每发现一个 SCC 时，就不断弹栈，直到这个 SCC 被完整弹出。
- 从 u 搜索到其儿子 v 时，考虑以下几种情况：
  - v 没被搜索过：直接进入 v 进行 dfs，然后令 $low[u] = \min(low[u], low[v])$；
  - v 被搜索过，已在栈中：令 $low[u] = \min(low[u], dfn[v])$；

Tarjan 算法

- 在该算法中，我们为每个节点 u 维护以下变量：
  - $dfn[u]$ 表示 dfs 时 u 是第几个被遍历到的；
  - $low[u]$ 表示 u 子树内跳到子树外的 dfn 最小值，初值等于 $dfn[u]$。
- 易见，一个子树内的 dfn 值都严格大于子树根。
- 在 dfs 的过程中，除了维护 dfn 和 low，我们需要将搜索到的点都放入一个栈内。随后，在每发现一个 SCC 时，就不断弹栈，直到这个 SCC 被完整弹出。
- 从 u 搜索到其儿子 v 时，考虑以下几种情况：
  - v 没被搜索过：直接进入 v 进行 dfs，然后令 $low[u] = \min(low[u], low[v])$；
  - v 被搜索过，已在栈中：令 $low[u] = \min(low[u], dfn[v])$；
  - v 被搜索过，已弹出栈：不用管。

Tarjan 算法

- 在该算法中，我们为每个节点 u 维护以下变量：
  - $dfn[u]$ 表示 dfs 时 u 是第几个被遍历到的；
  - $low[u]$ 表示 u 子树内跳到子树外的 dfn 最小值，初值等于 $dfn[u]$。
- 易见，一个子树内的 dfn 值都严格大于子树根。
- 在 dfs 的过程中，除了维护 dfn 和 low，我们需要将搜索到的点都放入一个栈内。随后，在每发现一个 SCC 时，就不断弹栈，直到这个 SCC 被完整弹出。
- 从 u 搜索到其儿子 v 时，考虑以下几种情况：
  - v 没被搜索过：直接进入 v 进行 dfs，然后令 $low[u] = \min(low[u], low[v])$；
  - v 被搜索过，已在栈中：令 $low[u] = \min(low[u], dfn[v])$；
  - v 被搜索过，已弹出栈：不用管。
- 而什么时候可以判定出现了一个 SCC？

Tarjan 算法

- 在该算法中，我们为每个节点 u 维护以下变量：
  - $dfn[u]$ 表示 dfs 时 u 是第几个被遍历到的；
  - $low[u]$ 表示 u 子树内跳到子树外的 dfn 最小值，初值等于 $dfn[u]$。
- 易见，一个子树内的 dfn 值都严格大于子树根。
- 在 dfs 的过程中，除了维护 dfn 和 low，我们需要将搜索到的点都放入一个栈内。随后，在每发现一个 SCC 时，就不断弹栈，直到这个 SCC 被完整弹出。
- 从 u 搜索到其儿子 v 时，考虑以下几种情况：
  - v 没被搜索过：直接进入 v 进行 dfs，然后令 $low[u] = \min(low[u], low[v])$；
  - v 被搜索过，已在栈中：令 $low[u] = \min(low[u], dfn[v])$；
  - v 被搜索过，已弹出栈：不用管。
- 而什么时候可以判定出现了一个 SCC？
- 当 $low[u]$ 计算完毕后，发现 $dfn[u] = low[u]$ 时，需要弹栈。

应用

应用

- 多次询问两个点是否可以互相到达；

应用

- 多次询问两个点是否可以互相到达；
- 求从一个点出发，可以到达多少个点（LOJ 10091）；

应用

- 多次询问两个点是否可以互相到达；
- 求从一个点出发，可以到达多少个点（LOJ 10091）；
- POJ 1236

应用

- 多次询问两个点是否可以互相到达；
- 求从一个点出发，可以到达多少个点（LOJ 10091）；
- POJ 1236
  - 求最少要将信息发送给多少个点，才能让它们的后继覆盖到所有点；

应用

- 多次询问两个点是否可以互相到达；
- 求从一个点出发，可以到达多少个点（LOJ 10091）；
- POJ 1236
  - 求最少要将信息发送给多少个点，才能让它们的后继覆盖到所有点；
  - 求最少添加几条边，才能使得无论信息发到哪一个点，都能被传递到所有点。

应用

- 多次询问两个点是否可以互相到达；
- 求从一个点出发，可以到达多少个点（LOJ 10091）；
- POJ 1236
  - 求最少要将信息发送给多少个点，才能让它们的后继覆盖到所有点；
  - 求最少添加几条边，才能使得无论信息发到哪一个点，都能被传递到所有点。
- 竞赛图的 SCC 结构

应用

- 多次询问两个点是否可以互相到达；
- 求从一个点出发，可以到达多少个点（LOJ 10091）；
- POJ 1236
  - 求最少要将信息发送给多少个点，才能让它们的后继覆盖到所有点；
  - 求最少添加几条边，才能使得无论信息发到哪一个点，都能被传递到所有点。
- 竞赛图的 SCC 结构
  - 一条链；

应用

- 多次询问两个点是否可以互相到达；
- 求从一个点出发，可以到达多少个点（LOJ 10091）；
- POJ 1236
  - 求最少要将信息发送给多少个点，才能让它们的后继覆盖到所有点；
  - 求最少添加几条边，才能使得无论信息发到哪一个点，都能被传递到所有点。
- 竞赛图的 SCC 结构
  - 一条链；
  - 哈密尔顿路；

应用

- 多次询问两个点是否可以互相到达；
- 求从一个点出发，可以到达多少个点（LOJ 10091）；
- POJ 1236
  - 求最少要将信息发送给多少个点，才能让它们的后继覆盖到所有点；
  - 求最少添加几条边，才能使得无论信息发到哪一个点，都能被传递到所有点。
- 竞赛图的 SCC 结构
  - 一条链；
  - 哈密尔顿路；
  - 如果一个竞赛图是强联通的，则其有哈密尔顿回路。

ABC 277 Ex

题目大意

问是否存在一个整数序列 $X_1, X_2, ..., X_n$ 满足如下条件，若存在需构造一组解：

- $0 \leq X_i \leq m, \forall 1 \leq i \leq n$
- $l_i \leq X_{a_i} + X_{b_i} \leq r_i, \forall 1 \leq i \leq q$

范围：$1 \leq n \leq 10000, 1 \leq m \leq 100, 1 \leq q \leq 10000$

ABC 277 Ex

题目大意

问是否存在一个整数序列 $X_1, X_2, ..., X_n$ 满足如下条件，若存在需构造一组解：

- $0 \le X_i \le m, \forall 1 \le i \le n$
- $l_i \le X_{a_i} + X_{b_i} \le r_i, \forall 1 \le i \le q$

范围：$1 \le n \le 10000, 1 \le m \le 100, 1 \le q \le 10000$

- 我们考虑 $n \times m$ 个 01 变量 $d_{i,j} = [X_i \ge j], \forall 1 \le i \le n, 1 \le j \le m$

ABC 277 Ex

题目大意

问是否存在一个整数序列 $X_1, X_2, ..., X_n$ 满足如下条件，若存在需构造一组解：

- $0 \le X_i \le m, \forall 1 \le i \le n$
- $l_i \le X_{a_i} + X_{b_i} \le r_i, \forall 1 \le i \le q$

范围：$1 \le n \le 10000, 1 \le m \le 100, 1 \le q \le 10000$

- 我们考虑 $n \times m$ 个 01 变量 $d_{i,j} = [X_i \ge j], \forall 1 \le i \le n, 1 \le j \le m$，有如下限制：

ABC 277 Ex

题目大意

问是否存在一个整数序列 $X_1, X_2, ..., X_n$ 满足如下条件，若存在需构造一组解：

- $0 \le X_i \le m, \forall 1 \le i \le n$
- $l_i \le X_{a_i} + X_{b_i} \le r_i, \forall 1 \le i \le q$

范围：$1 \le n \le 10000, 1 \le m \le 100, 1 \le q \le 10000$

- 我们考虑 $n \times m$ 个 01 变量 $d_{i,j} = [X_i \ge j], \forall 1 \le i \le n, 1 \le j \le m$，有如下限制：
  - 禁止 $d_{i,j} = 0, d_{i,j+1} = 1$

ABC 277 Ex

题目大意

问是否存在一个整数序列 $X_1, X_2, ..., X_n$ 满足如下条件，若存在需构造一组解：

- $0 \le X_i \le m, \forall 1 \le i \le n$
- $l_i \le X_{a_i} + X_{b_i} \le r_i, \forall 1 \le i \le q$

范围：$1 \le n \le 10000, 1 \le m \le 100, 1 \le q \le 10000$

- 我们考虑 $n \times m$ 个 01 变量 $d_{i,j} = [X_i \ge j], \forall 1 \le i \le n, 1 \le j \le m$，有如下限制：
  - 禁止 $d_{i,j} = 0, d_{i,j+1} = 1$
  - 对于一组限制 $l \le X_a + X_b \le r$，枚举 $1 \le i \le m$：

ABC 277 Ex

题目大意

问是否存在一个整数序列 $X_1, X_2, ..., X_n$ 满足如下条件，若存在需构造一组解：

- $0 \le X_i \le m, \forall 1 \le i \le n$
- $l_i \le X_{a_i} + X_{b_i} \le r_i, \forall 1 \le i \le q$

范围：$1 \le n \le 10000, 1 \le m \le 100, 1 \le q \le 10000$

- 我们考虑 $n \times m$ 个 01 变量 $d_{i,j} = [X_i \ge j], \forall 1 \le i \le n, 1 \le j \le m$，有如下限制：
  - 禁止 $d_{i,j} = 0, d_{i,j+1} = 1$
  - 对于一组限制 $l \le X_a + X_b \le r$，枚举 $1 \le i \le m$：
    - 若 $r < i$，则禁止 $d_{a,i} = 1$；否则，$d_{a,i} = 1 \Rightarrow d_{b,r-i+1} = 0$

ABC 277 Ex

题目大意

问是否存在一个整数序列 $X_1, X_2, ..., X_n$ 满足如下条件，若存在需构造一组解：

- $0 \le X_i \le m, \forall 1 \le i \le n$
- $l_i \le X_{a_i} + X_{b_i} \le r_i, \forall 1 \le i \le q$

范围：$1 \le n \le 10000, 1 \le m \le 100, 1 \le q \le 10000$

- 我们考虑 $n \times m$ 个 01 变量 $d_{i,j} = [X_i \ge j], \forall 1 \le i \le n, 1 \le j \le m$，有如下限制：
  - 禁止 $d_{i,j} = 0, d_{i,j+1} = 1$
  - 对于一组限制 $l \le X_a + X_b \le r$，枚举 $1 \le i \le m$：
    - 若 $r < i$，则禁止 $d_{a,i} = 1$；否则，$d_{a,i} = 1 \Rightarrow d_{b,r-i+1} = 0$
    - 若 $l - (i-1) > m$，则禁止 $d_{a,i} = 0$；否则，$d_{a,i} = 0 \Rightarrow d_{b,l-(i-1)} = 1$

ABC 277 Ex

题目大意

问是否存在一个整数序列 $X_1, X_2, ..., X_n$ 满足如下条件，若存在需构造一组解：

- $0 \le X_i \le m, \forall 1 \le i \le n$
- $l_i \le X_{a_i} + X_{b_i} \le r_i, \forall 1 \le i \le q$

范围：$1 \le n \le 10000, 1 \le m \le 100, 1 \le q \le 10000$

- 我们考虑 $n \times m$ 个 01 变量 $d_{i,j} = [X_i \ge j], \forall 1 \le i \le n, 1 \le j \le m$，有如下限制：
  - 禁止 $d_{i,j} = 0, d_{i,j+1} = 1$
  - 对于一组限制 $l \le X_a + X_b \le r$，枚举 $1 \le i \le m$:
    - 若 $r < i$，则禁止 $d_{a,i} = 1$；否则，$d_{a,i} = 1 \Rightarrow d_{b,r-i+1} = 0$
    - 若 $l - (i-1) > m$，则禁止 $d_{a,i} = 0$；否则，$d_{a,i} = 0 \Rightarrow d_{b,l-(i-1)} = 1$
- 跑 2-SAT，并通过比较 $d_{i,j} = 0$ 与 $d_{i,j} = 1$ 在 SCC 缩点后拓扑序先后顺序，判断 $d_{i,j}$ 的实际取值为 0 还是 1，从而输出方案。

割点与点双联通分量

割点与点双联通分量

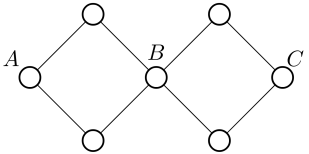
割点

- 连通无向图中，若一个点被删除后连通块个数增加，则称这个点为割点。

割点与点双联通分量

割点

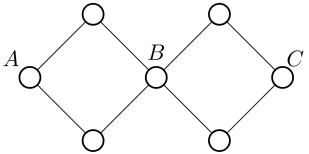
- 连通无向图中，若一个点被删除后连通块个数增加，则称这个点为割点。



割点与点双联通分量

割点

- 连通无向图中，若一个点被删除后连通块个数增加，则称这个点为割点。

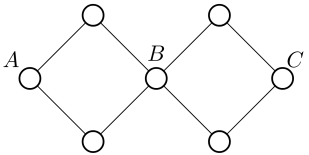


点双连通分量

割点与点双联通分量

割点

- 连通无向图中，若一个点被删除后连通块个数增加，则称这个点为割点。



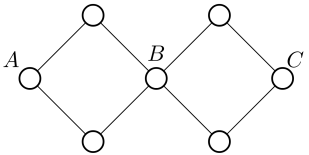
点双连通分量

- 连通无向图中，对于两个点 x, y，若无论删去哪个其他点（除了 u, v）它们都仍然连通，则称 u, v 点双连通。

割点与点双联通分量

割点

- 连通无向图中，若一个点被删除后连通块个数增加，则称这个点为割点。



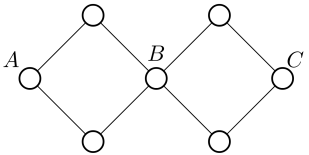
点双连通分量

- 连通无向图中，对于两个点 x, y，若无论删去哪个其他点（除了 u, v）它们都仍然连通，则称 u, v 点双连通。
- 点双没有传递性（如上图），但是仍可以定义并求出所有极大的点双连通分量。

割点与点双联通分量

割点

- 连通无向图中，若一个点被删除后连通块个数增加，则称这个点为割点。



点双连通分量

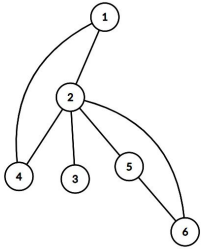
- 连通无向图中，对于两个点 x, y，若无论删去哪个其他点（除了 u, v）它们都仍然连通，则称 u, v 点双连通。
- 点双没有传递性（如上图），但是仍可以定义并求出所有极大的点双连通分量。
- 一个几乎等价的定义是不存在割点的图，但特别的情况是一个点双只有 2 个点。

利用 tarjan 求割点与点双

- $dfn[u]$：dfs 序；$low[u]$ 为 u 子树内走非树边可以到达的 dfn 最小值。

利用 tarjan 求割点与点双

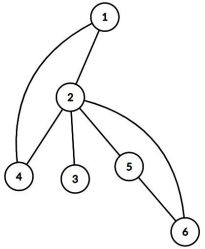
- $dfn[u]$：dfs 序；$low[u]$ 为 u 子树内走非树边可以到达的 dfn 最小值。



- e.g. $low[2] = 1, low[5] = 2$

利用 tarjan 求割点与点双

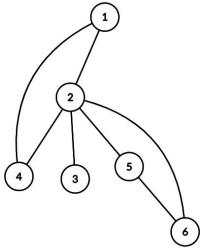
- $dfn[u]$：dfs 序；$low[u]$ 为 u 子树内走非树边可以到达的 dfn 最小值。



- e.g. $low[2] = 1, low[5] = 2$
- **割点**：当 u 是 v 父亲，$low[v] \geq dfn[u]$ 时，u 即为割点。

利用 tarjan 求割点与点双

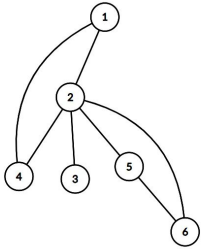
- $dfn[u]$：dfs 序；$low[u]$ 为 u 子树内走非树边可以到达的 dfn 最小值。



- e.g. $low[2] = 1, low[5] = 2$
- 割点：当 u 是 v 父亲，$low[v] \geq dfn[u]$ 时，u 即为割点。
- 点双：在 dfs 时，将经过的树边都加入栈；遇到一个 u, v 符合割点条件时，就将栈里的边依次弹出，构成一个新点双，直到弹到 $u - v$ 这条边。

利用 tarjan 求割点与点双

- $dfn[u]$：dfs 序；$low[u]$ 为 u 子树内走非树边可以到达的 dfn 最小值。



- e.g. $low[2] = 1, low[5] = 2$
- 割点：当 u 是 v 父亲，$low[v] \geq dfn[u]$ 时，u 即为割点。
- 点双：在 dfs 时，将经过的树边都加入栈；遇到一个 u, v 符合割点条件时，就将栈里的边依次弹出，构成一个新点双，直到弹到 $u - v$ 这条边。
- 所有点双连通分量只会在割点处相交。

割边与边双联通分量

割边与边双联通分量

割边

连通无向图中，如果一条边被删除，连通块数量增加，则称其为割边。

割边与边双联通分量

割边

连通无向图中，如果一条边被删除，连通块数量增加，则称其为割边。

边双连通分量

- 连通无向图中，对于两个点 x, y，若无论删去哪条边，它们都仍然连通，则称 u, v 边双连通。

割边与边双联通分量

割边

连通无向图中，如果一条边被删除，连通块数量增加，则称其为割边。

边双连通分量

- 连通无向图中，对于两个点 x, y，若无论删去哪条边，它们都仍然连通，则称 u, v 边双连通。
- 边双有传递性。所有极大边双连通分量可以通过删去所有割边获得。

利用 tarjan 求割边

利用 tarjan 求割边

- 与求割点完全一致地定义：

利用 tarjan 求割边

- 与求割点完全一致地定义：
- $dfn[u]$：dfs 序；$low[u]$ 为 u 子树内走非树边可以到达的 dfn 最小值。

利用 tarjan 求割边

- 与求割点完全一致地定义：
- $dfn[u]$：dfs 序；$low[u]$ 为 u 子树内走非树边可以到达的 dfn 最小值。
- 割边：当 u 是 v 父亲，$low[v] = dfn[v]$ 时，u 到 v 的边即为割边。

利用 tarjan 求割边

- 与求割点完全一致地定义：
- $dfn[u]$：dfs 序；$low[u]$ 为 u 子树内走非树边可以到达的 dfn 最小值。
- 割边：当 u 是 v 父亲，$low[v] = dfn[v]$ 时，u 到 v 的边即为割边。
- 而非树边显然不会是割边。

POJ 1523

题目大意

给定一张连通无向图，问删除每个点之后会有几个连通块。

POJ 1523

题目大意

给定一张连通无向图，问删除每个点之后会有几个连通块。

对于 u 来说，每个 $low[v] \geq dfn[u]$ 的儿子 v 都会在删 u 之后，变成新的连通块。

POJ 1523

题目大意

给定一张连通无向图，问删除每个点之后会有几个连通块。

对于 u 来说，每个 $low[v] \geq dfn[u]$ 的儿子 v 都会在删 u 之后，变成新的连通块。此外，求割点的具体实现中需特判 u 是 dfs 树根的情况，此时 u 是割点只与儿子个数是否 > 1 有关。

ABC 301 Ex

题目大意

给定一张 n 个点 m 条边的连通无向图，带边权，定义 $d(s,t)$ 为 s 到 t 的最大边权最小值。
回答 q 次询问，每次询问形如 a,s,t，你需要回答：将第 a 条边的边权 $+1$ 之后，$d(s,t)$ 会增加多少？
注意询问是独立的，即不会真的修改边权。
范围：$n,m,q\leq 2\times 10^5$

ABC 301 Ex

题目大意

给定一张 n 个点 m 条边的连通无向图，带边权，定义 $d(s,t)$ 为 s 到 t 的最大边权最小值。
回答 q 次询问，每次询问形如 a,s,t，你需要回答：将第 a 条边的边权 $+1$ 之后，$d(s,t)$ 会增加多少？
注意询问是独立的，即不会真的修改边权。
范围：$n,m,q\le 2\times 10^5$

- 首先，$d(s,t)\neq w_a$ 时，增加 w_a 不会影响 $d(s,t)$。

ABC 301 Ex

题目大意

给定一张 n 个点 m 条边的连通无向图，带边权，定义 $d(s,t)$ 为 s 到 t 的最大边权最小值。
回答 q 次询问，每次询问形如 a,s,t，你需要回答：将第 a 条边的边权 $+1$ 之后，$d(s,t)$ 会增加多少？
注意询问是独立的，即不会真的修改边权。
范围：$n,m,q\le 2\times 10^5$

- 首先，$d(s,t)\neq w_a$ 时，增加 w_a 不会影响 $d(s,t)$。而检查 $d(s,t)=w_a$ 只需检查 s,t 在分别加入所有 $\le w_a-1$ 和 $\le w_a$ 的边后的连通情况，容易离线处理。

ABC 301 Ex

题目大意

给定一张 n 个点 m 条边的连通无向图，带边权，定义 $d(s,t)$ 为 s 到 t 的最大边权最小值。
回答 q 次询问，每次询问形如 a,s,t，你需要回答：将第 a 条边的边权 $+1$ 之后，$d(s,t)$ 会增加多少？
注意询问是独立的，即不会真的修改边权。
范围：$n,m,q\le 2\times 10^5$

- 首先，$d(s,t)\neq w_a$ 时，增加 w_a 不会影响 $d(s,t)$。而检查 $d(s,t)=w_a$ 只需检查 s,t 在分别加入所有 $\le w_a-1$ 和 $\le w_a$ 的边后的连通情况，容易离线处理。
- 其次，若 a 不是割边，则增加 w_a 也不会影响 $d(s,t)$，因为完全可以绕开 a。

ABC 301 Ex

题目大意

给定一张 n 个点 m 条边的连通无向图，带边权，定义 $d(s,t)$ 为 s 到 t 的最大边权最小值。
回答 q 次询问，每次询问形如 a,s,t，你需要回答：将第 a 条边的边权 $+1$ 之后，$d(s,t)$ 会增加多少？
注意询问是独立的，即不会真的修改边权。
范围：$n,m,q\le 2\times 10^5$

- 首先，$d(s,t)\neq w_a$ 时，增加 w_a 不会影响 $d(s,t)$。而检查 $d(s,t)=w_a$ 只需检查 s,t 在分别加入所有 $\le w_a-1$ 和 $\le w_a$ 的边后的连通情况，容易离线处理。
- 其次，若 a 不是割边，则增加 w_a 也不会影响 $d(s,t)$，因为完全可以绕开 a。
- 下设 $d(s,t)=w_a$ 且 a 是割边。则我们只要判断 a 是否在 s 到 t 的必由之路上，即断开 a 是否会影响 s 到 t 的连通性。

ABC 301 Ex

题目大意

给定一张 n 个点 m 条边的连通无向图，带边权，定义 $d(s,t)$ 为 s 到 t 的最大边权最小值。
回答 q 次询问，每次询问形如 a,s,t，你需要回答：将第 a 条边的边权 $+1$ 之后，$d(s,t)$ 会增加多少？
注意询问是独立的，即不会真的修改边权。
范围：$n,m,q \leq 2\times 10^5$

- 首先，$d(s,t)\neq w_a$ 时，增加 w_a 不会影响 $d(s,t)$。而检查 $d(s,t)=w_a$ 只需检查 s,t 在分别加入所有 $\leq w_a-1$ 和 $\leq w_a$ 的边后的连通情况，容易离线处理。
- 其次，若 a 不是割边，则增加 w_a 也不会影响 $d(s,t)$，因为完全可以绕开 a。
- 下设 $d(s,t)=w_a$ 且 a 是割边。则我们只要判断 a 是否在 s 到 t 的必由之路上，即断开 a 是否会影响 s 到 t 的连通性。
- 这容易通过边双缩点成为一棵树后，判断祖先后代关系来解决，具体比较 DFS 序就可以了。

引入

引入

- 最初的圆方树是用于简化仙人掌结构的，比如处理仙人掌 dp。不过是否显式地建立出圆方树并不是很关键。

引入

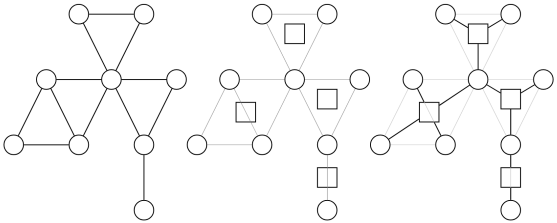
- 最初的圆方树是用于简化仙人掌结构的，比如处理仙人掌 dp。不过是否显式地建立出圆方树并不是很关键。
- 这里我们讨论的圆方树是基于点双的，也就是从仙人掌推广到了一般的连通无向图。

引入

- 最初的圆方树是用于简化仙人掌结构的，比如处理仙人掌 dp。不过是否显式地建立出圆方树并不是很关键。
- 这里我们讨论的圆方树是基于点双的，也就是从仙人掌推广到了一般的连通无向图。
- 我们对于每个点双新建一个方点，连接所有该点双内的点。原图中的点称作圆点，原图中的边不用添加。

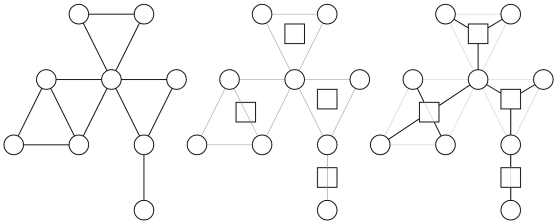
引入

- 最初的圆方树是用于简化仙人掌结构的，比如处理仙人掌 dp。不过是否显式地建立出圆方树并不是很关键。
- 这里我们讨论的圆方树是基于点双的，也就是从仙人掌推广到了一般的连通无向图。
- 我们对于每个点双新建一个方点，连接所有该点双内的点。原图中的点称作圆点，原图中的边不用添加。



引入

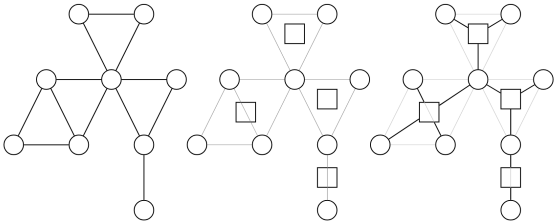
- 最初的圆方树是用于简化仙人掌结构的，比如处理仙人掌 dp。不过是否显式地建立出圆方树并不是很关键。
- 这里我们讨论的圆方树是基于点双的，也就是从仙人掌推广到了一般的连通无向图。
- 我们对于每个点双新建一个方点，连接所有该点双内的点。原图中的点称作圆点，原图中的边不用添加。



- 由此得到一棵树，每条边恰好连接一对方点和圆点，我们称作圆方树。

引入

- 最初的圆方树是用于简化仙人掌结构的，比如处理仙人掌 dp。不过是否显式地建立出圆方树并不是很关键。
- 这里我们讨论的圆方树是基于点双的，也就是从仙人掌推广到了一般的连通无向图。
- 我们对于每个点双新建一个方点，连接所有该点双内的点。原图中的点称作圆点，原图中的边不用添加。



- 由此得到一棵树，每条边恰好连接一对方点和圆点，我们称作圆方树。
- 需注意圆方树的节点个数最多为原图的两倍，所以需要开大数组。

CF 487 E Tourists

题目大意

给定一张 n 个点 m 条边的连通无向图，带点权。有 q 次询问，形如：

- 1 a w，表示将点 a 的点权修改为 w；
- 2 a b，表示 tourist 将从 a 走到 b，在不走重复点的情况下，他可以经过的最小点权是多少。

范围：$1 \leq n, m, q \leq 10^5$。

CF 487 E Tourists

- 看到点权 + 不走重复点，容易想到点双缩点，因此我们建立圆方树。

CF 487 E Tourists

- 看到点权 + 不走重复点，容易想到点双缩点，因此我们建立圆方树。
- 问题变为每次修改圆点权值，查询树上路径最小值。方点的权值我们定义为其邻居的点权最小值。

CF 487 E Tourists

- 看到点权 + 不走重复点，容易想到点双缩点，因此我们建立圆方树。
- 问题变为每次修改圆点权值，查询树上路径最小值。方点的权值我们定义为其邻居的点权最小值。
- 我们当然可以暴力修改方点的权值，但是当一个割点同时处在很多点双中，复杂度不可接受。

CF 487 E Tourists

- 看到点权 + 不走重复点，容易想到点双缩点，因此我们建立圆方树。
- 问题变为每次修改圆点权值，查询树上路径最小值。方点的权值我们定义为其邻居的点权最小值。
- 我们当然可以暴力修改方点的权值，但是当一个割点同时处在很多点双中，复杂度不可接受。
- 但事实上，方点没有必要记录其父亲圆点的权值，因为一条路径经过了方点，也大概率会直接经过其父亲。

CF 487 E Tourists

- 看到点权 + 不走重复点，容易想到点双缩点，因此我们建立圆方树。
- 问题变为每次修改圆点权值，查询树上路径最小值。方点的权值我们定义为其邻居的点权最小值。
- 我们当然可以暴力修改方点的权值，但是当一个割点同时处在很多点双中，复杂度不可接受。
- 但事实上，方点没有必要记录其父亲圆点的权值，因为一条路径经过了方点，也大概率会直接经过其父亲。
- 因此我们将方点的权值定义为儿子的点权最小值，通过在方点处维护 multiset，可以 $O(\log)$ 处理修改。

CF 487 E Tourists

- 看到点权 + 不走重复点，容易想到点双缩点，因此我们建立圆方树。
- 问题变为每次修改圆点权值，查询树上路径最小值。方点的权值我们定义为其邻居的点权最小值。
- 我们当然可以暴力修改方点的权值，但是当一个割点同时处在很多点双中，复杂度不可接受。
- 但事实上，方点没有必要记录其父亲圆点的权值，因为一条路径经过了方点，也大概率会直接经过其父亲。
- 因此我们将方点的权值定义为儿子的点权最小值，通过在方点处维护 multiset，可以 $O(\log)$ 处理修改。
- 随后使用树链剖分 + 线段树来解决路径最小值，最后注意特判 LCA 是方点的情况，此时需额外加入其父亲。

洛谷 4606 [SDOI 2018] 战略游戏

题目大意

给定一张 n 个点 m 条边的连通无向图，每次询问给出一个点集 S，回答有多少个 $u \notin S$，满足删除 u 后，S 中存在至少两个点不再连通。

范围：$2 \leq |S| \leq n \leq 10^5, \sum |S| \leq 2 \times 10^5$

洛谷 4606 [SDOI 2018] 战略游戏

题目大意

给定一张 n 个点 m 条边的连通无向图，每次询问给出一个点集 S，回答有多少个 $u \notin S$，满足删除 u 后，S 中存在至少两个点不再连通。
范围：$2 \leq |S| \leq n \leq 10^5, \sum |S| \leq 2 \times 10^5$

- 两点之间必经的点就是圆方树路径上的全体圆点。

洛谷 4606 [SDOI 2018] 战略游戏

题目大意

给定一张 n 个点 m 条边的连通无向图，每次询问给出一个点集 S，回答有多少个 $u \notin S$，满足删除 u 后，S 中存在至少两个点不再连通。
范围：$2 \leq |S| \leq n \leq 10^5, \sum |S| \leq 2 \times 10^5$

- 两点之间必经的点就是圆方树路径上的全体圆点。
- 因此对于读入的集合 S，求出其在圆方树上的虚树，我们只要计算该虚树内圆点个数 $-|S|$ 即可。

洛谷 4606 [SDOI 2018] 战略游戏

题目大意

给定一张 n 个点 m 条边的连通无向图，每次询问给出一个点集 S，回答有多少个 $u \notin S$，满足删除 u 后，S 中存在至少两个点不再连通。
范围：$2 \leq |S| \leq n \leq 10^5, \sum |S| \leq 2 \times 10^5$

- 两点之间必经的点就是圆方树路径上的全体圆点。
- 因此对于读入的集合 S，求出其在圆方树上的虚树，我们只要计算该虚树内圆点个数 $-|S|$ 即可。有两种办法：

洛谷 4606 [SDOI 2018] 战略游戏

题目大意

给定一张 n 个点 m 条边的连通无向图，每次询问给出一个点集 S，回答有多少个 $u \notin S$，满足删除 u 后，S 中存在至少两个点不再连通。
范围：$2 \leq |S| \leq n \leq 10^5, \sum |S| \leq 2 \times 10^5$

- 两点之间必经的点就是圆方树路径上的全体圆点。
- 因此对于读入的集合 S，求出其在圆方树上的虚树，我们只要计算该虚树内圆点个数 $-|S|$ 即可。有两种办法：
- 1. 建立虚树，统计每个点到虚树父亲路径上的圆点数量；

洛谷 4606 [SDOI 2018] 战略游戏

题目大意

给定一张 n 个点 m 条边的连通无向图，每次询问给出一个点集 S，回答有多少个 $u \notin S$，满足删除 u 后，S 中存在至少两个点不再连通。
范围：$2 \le |S| \le n \le 10^5, \sum |S| \le 2 \times 10^5$

- 两点之间必经的点就是圆方树路径上的全体圆点。
- 因此对于读入的集合 S，求出其在圆方树上的虚树，我们只要计算该虚树内圆点个数 $-|S|$ 即可。有两种办法：
- 1. 建立虚树，统计每个点到虚树父亲路径上的圆点数量；
- 2. 不直接建立虚树，将 S 内的点按 DFS 序排序，统计 DFS 序相邻的路径上圆点数量，然后除以 2。注意相邻的也包括头尾。

课后习题

- HDU 3639
- HDU 3072
- ABC 295 G
- HDU 2460
- CF 1697 F（与 ABC 277 Ex 基本一致）

谢谢大家！